Navegação Autónoma de Robots: Interpretação dos Dados Sensoriais e Navegação Local

Vítor Manuel Ferreira dos Santos

Universidade de Aveiro

Julho 1995

Tabela de conteúdo

Resumo

O trabalho descrito nesta tese foca os problemas da navegação local de robots móveis. A ideia fundamental consiste em desenvolver mapas de percepção à base de dados de ultra-som usando redes neuronais e em seguida efectuar navegação local com base nesses mapas. A tarefa de navegação local é ainda integrada numa arquitectura global onde se definem outras tarefas para poder levar a cabo operações completas de navegação.

As capacidades de integração e combinação de informação das redes neuronais são aproveitadas para eliminar ou reduzir alguns problemas associados à medição com sensores de ultra-som, nomeadamente as reflexões especulares que resultam em falsas medições. As redes geram assim os mapas de percepção que são em seguida usados para efectuar a navegação local.

Os mapas de percepção definidos são especiais no modo em que se adequam às características da medição com sensores de ultra-som, nomeadamente a forma geométrica do feixe de medição, e a fiabilidade das medições com a distância e orientação. Este tipo de mapa destaca-se de dois modos dos conceitos tradicionais no mapeamento de ambientes em robótica. Em primeiro lugar, o mapa é solidário com o robot, logo é referido ao próprio robot e não ao ambiente onde se desloca. Em segundo lugar, estes mapas de percepção não são baseados em grelhas Cartesianas, mas sim em grelhas com características radiais.

A navegação local de um robot consiste em executar movimento de uma forma segura dentro do espaço percebido do ponto de vista do robot, e representado no mapa de percepção, obedecendo a uma dada estratégia ou comportamento em vigor no momento, mas sem qualquer referência a localização ou posicionamento dentro do ambiente onde se move. O módulo responsável pela navegação local está integrado numa arquitectura global de navegação, mas é de igual modo uma entidade autónoma que por si só pode proporcionar um meio de deambulação do robot, isto é, navegação sem rumo ou destino pré-definido.

A arquitectura global de navegação conta com vários módulos entre os quais se destaca também o detector de colisões iminentes que se articula de perto com o módulo de navegação local em situações de ambientes muito densos de obstáculos. A arquitectura está concebida de forma a poder incluir facilmente outros módulos já existentes, como, por exemplo, um módulo de localização ou de planeamento de trajectória a alto nível. Todavia, outros módulos são especiais e tem de ser desenvolvidos para esta arquitectura como é o caso de um seguidor de trajectórias que, embora mantendo o seu nível de modularidade, servirá de ligação entre o planeamento de uma trajectória ou um destino a atingir, e a sua execução a mais baixo nível incluindo o desvio de obstáculos desconhecidos ou imprevistos.

Abstract

This thesis focuses on the problems of local navigation of mobile robots. The main idea consists on the development of ultrasonic perception maps, by means of neural networks, and to perform local navigation based on those maps. The local navigation task is integrated in a global architecture along with some other tasks to allow complete navigation operations.

The neural network data integration and combination abilities are explored to eliminate or reduce some of the problems associated to the ranging with ultrasonic devices, namely the specular reflections that yield false measurements. The neural nets generate the perception maps after the raw ultrasound data.

The perception maps are special in the way that they conform to the properties of the physical principles of ultrasonic ranging, namely the shape of measuring beams, and the reliability of the measurements with distance and orientation. This type of map is not bound to the traditional concepts on environment mapping in robotics in two distinct ways. First, because the map is in the frame of the robot, therefore moving as the robot moves. Second, it is not based on a Cartesian grid; a radial-like grid is used instead.

Local navigation is here defined as the ability of performing safe motion within the free space perceived around the robot, obeying to some strategy or behaviour, but without any references to localisation or positioning in the working environment. The local navigation module is part of a global architecture for navigation, as well as a standalone mean of performing robot deambulation in the environment.

The full navigation architecture consists of several modules. One of the most important is an eminent collision detection component, closely articulated with the local navigation module in case of very difficult (cluttered) environments. The architecture is conceived in a way that already existent modules can be easily integrated, such as localisation or high level path planning. However, some other modules are specially conceived for this architecture, such as the path follower that, although keeping its level of modularity, will establish the connection between a high level planned path or target to reach, and its execution at a lower level, including, but not limited to, obstacle avoidance.

Introdução

0.1. Motivação e objectivos

O trabalho descrito nesta tese pretende propôr uma solução, ou melhoramentos às soluções existentes para alguns problemas relacionados com a robótica móvel. As questões abordadas situam-se principalmente nos domínios da navegação e da problemática associada aos sensores de ultra-som.

O principal objectivo estipulado para este trabalho foi o de definir técnicas de navegação mais ou menos autónoma com base em sensores exclusivamente a bordo, e com principal destaque para os sensores de ultra-som.

Sendo bastante conhecidos os problemas associados a este tipo de sensores, nomeadamente no que respeita à interpretação das medidas que fornecem, a principal componente de investigação foi precisamente a questão de definir uma alternativa para a interpretação, processamento e utilização dos dados de ultra-som. O *leitmotiv* na procura deste método foram as redes neuronais: os seus resultados passados em problemas de difícil modelização e a necessidade aparente de um instrumento combinador de informação complexa e dispersa, como é o caso dos ultra-sons, foram a justificação para esta escolha, que se mostrou válida e vantajosa quando comparada com outras alternativas, como se verá.

Contudo, a percepção é apenas uma parte do processo de navegação: a segunda grande empresa foi o de conseguir efectuar navegação autónoma a partir das novas representações obtidas dos dados sensoriais. Estas representações têm sobretudo a ver com a distribuição de espaço livre em torno do robot, e foi então necessário definir novos conceitos de navegação. A primeira separação da maioria das linhas tradicionais foi a de conceber a navegação local como um processo independente do ambiente onde se navega e sem qualquer conhecimento *a priori*. Tendo em mão uma percepção do espaço livre, tratou-se depois tão só de definir estratégias de navegação para se chegar de facto à locomoção do robot pelos seus próprios meios.

0.2. Estrutura da tese

Este texto está dividido em três partes principais: a primeira expõe as diferentes componentes da navegação, os métodos existentes para levar a cabo as tarefas inerentes a cada uma delas e por fim os problemas que ainda não estão resolvidos ou podem ainda ser

melhorados. A primeira parte explica ainda os sensores de ultra-som usados neste trabalho, os mais usados aliás, e quais os seus problemas com que se debatem ainda os investigadores.

A segunda parte está escrita em forma de proposta de solução para os problemas expostos na primeira parte. Consiste exactamente em definir conceitos adequados aos problemas, e daí encaminhar-se para uma solução para a navegação autónoma, semi-autónoma ou assistida de um robot móvel.

Finalmente, a terceira parte detalha os resultados experimentais obtidos. Discute ainda a possível validade dos métodos e técnicas noutras circunstâncias, isto é, para outros robots.

Em complemento apresenta-se ainda um conjunto de apêndices a fornecer informação sobre algumas componentes e ferramentas do trabalho tal como noções básicas sobre redes neuronais, alguns algoritmos usados e descrição do ambiente de trabalho, entre outros.

0.3. Contribuições originais

As principais contribuições originais deste trabalho resumem-se concisamente no seguinte:

- Separação clara e radical da **navegação local** no processo de navegação de um robot móvel.
- Concepção de **mapas de percepção** dedicados à representação de dados de ultra-som.
- Uso de **redes neuronais** para construir representações de espaço livre em redor do robot usando dados de ultra-som.
- Introdução de progressos em **autonomia** e **navegação assistida**.

A primeira das contribuições resume todo um conjunto de novidades introduzidas relativamente aos sistemas tradicionais: houve toda uma modularização das tarefas de navegação, e foi definida uma arquitectura com vários níveis de controlo (**Baixo nível** — emergências, **Médio nível** — navegação local, **Alto nível** — trajectórias e tarefas completas de navegação). A navegação local acarretou um enaltecimento do espaço em torno do robot, contrariamente aos sistemas tradicionais que procuram incessantemente um mapa global e a localização do robot, e não evocam tão frequentemente a importância da quantidade de espaço livre local em todas as direcções.

A segunda grande componente das contribuições originais envolve um estudo da natureza das medições com sensores de ultra-som e na consequente concepção de uma representação dedicada do espaço. É em si já pouco usual procurar representar o espaço visto do robot, mas surge inédito conceber uma estrutura com a topologia das **grelhas de geometria geral** para fazer essa representação. A introdução do mapa de percepção permitiu uma redução da complexidade sensorial, e forneceu ao mesmo tempo um elemento de grande ajuda para o desenvolvimento da própria navegação local.

Finalmente, a última contribuição maior foi o recorrer ao uso de redes neuronais para construir a representação do espaço livre através dos mapas de percepção. A opção das redes,

mais do que o seguir de uma tendência generalizada nos últimos anos nas áreas da investigação científica, foi uma aposta algo arriscada dada a pouca maturidade que estes sistemas ainda apresentam. A ideia fazia contudo sentido, dadas no início as necessidades aparentes de um sistema capaz de lidar com dados falíveis, e ser capaz ao mesmo tempo de aproveitar a redundância de informação que se sabia existir no sistema sensorial.

Em suma, a contribuição global traduz-se afinal na introdução com sucesso de progressos em direcção à autonomia de robot móveis e, também importante, naquilo a que se chamou navegação assistida, que permite uma simbiose entre o comando manual e o comando automático do robot móvel, aumentando a segurança e o apoio à navegação humanamente controlada.

Parte 1

Dualidade da Questão da Robótica Móvel: Navegação e Sensores.

1.1. Introdução

A perspectiva humana sobre a robótica tem-se adaptado de tal modo que, hoje em dia, os idealismos de máquinas universais deram lugar aos conformismos de repartição de objectivos e especialização de tarefas. Até o antropomorfismo cedeu a estruturas mais práticas e adequadas a situações da vida real: quem pela primeira vez admirava os autómatos com forma humana nas feiras da década de 50, decerto que exprimiria pelo menos uma certa desilusão perante, por exemplo, os robots para inspecção interna de canos, desenvolvidos nos últimos anos pela indústria japonesa! A tendência andróide está a passar inevitavelmente por uma fase recessiva, quiçá, para numa próxima fase, já com a tecnologia e o conhecimento suficientemente desenvolvidos, se poder retomá-la.

O termo **robótica** surge como o nome da disciplina que estuda os aspectos relacionados com *robots*, mais concretamente: o projecto, a construção e as aplicações. O estrangeirismo *robot* deriva da palavra checa *'ròbota'* que significa trabalho pesado ou forçado, ou analogamente da palavra *'robotnik'* que ainda em checo significa servo. A popularidade deste termo parte de uma peça do checoslovaco Karel Čapek datada de 1921, onde as máquinas substituíam os homens no trabalho, e acabavam por eliminar os seus patrões e dominar o mundo. É ainda de salientar que partir dos anos 40 o escritor Isaac Asimov (1920-1992) divulgou nas suas obras literárias, mas sempre com grande preocupação científica, estes conceitos de **robótica** e **robot**, para os quais definiu inclusive as famosas **Leis da Robótica** que, aliás, reviu e completou em 1985 [Clarke93, Clarke94].

Um possível termo da língua portuguesa para designar *robot* é a palavra **autómato** que, todavia, além de possuir menor expressividade, é hoje em dia usada para representar uma mais vasta categoria de máquinas destinadas a executar ou facilitar o trabalho de seres humanos com pouca ou nenhuma supervisão destes. Os robots são assim um subgrupo de dispositivos automáticos especializados, dotados de maior autonomia e capacidade de serem programados, resultando deste modo em sistemas mais versáteis que os "normais" autómatos cuja sequência de acções é geralmente simples e invariável no tempo e no espaço (como os simples brinquedos animados a corda ou a pilhas para as crianças, ou até algumas complexas máquinas ferramenta das instalações industriais).

Outras definições de robot/robótica que se encontram, incluem as seguintes:

O texto da lista de questões frequentes (*FAQ-Frequently Asked Questions*) do grupo *comp.robotics* da *Usenet* sugere entre outras:

> "Dispositivos electro-mecânicos pré-programáveis para execução de uma variedade de funções."

O consagrado dicionário Webster propõe:

> "Dispositivo automático que executa funções normalmente atribuídas a humanos ou uma máquina com a forma de um humano."

Em 1986, Phillip H. McKerrow propôs para robot a seguinte definição:

> "Um robot é uma máquina que pode ser programada para fazer uma variedade de tarefas, do mesmo modo que um computador é um circuito electrónico que pode ser programado para fazer uma variedade de tarefas."

E, continuando a citar McKerrow, temos de seguida uma definição de robótica com um sabor algo recursivo [repare-se em a) e d)] :

> "Robótica é a disciplina que envolve: a) o projecto, construção, controlo e programação de robots; b) o uso de robots para resolver problemas; c) o estudo dos processos de controlo, sensores e algoritmos usados em humanos, animais e máquinas, e; d) a aplicação destes processos de controlo e destes algoritmos para o projecto de robots."

A robótica é uma disciplina que envolve a cooperação de inúmeras áreas da ciência e tecnologia, e frequentemente exige delas os seus maiores progressos, como por exemplo é o caso da visão artificial. Contudo, pareceres optimistas como o de Joseph F. Engelberger, uma personalidade da indústria robótica Norte-Americana e considerado um dos fundadores da robótica industrial, afirmam que já se disporá dos meios mecânicos, sensoriais e computacionais para construir robots adequados à maioria das tarefas (sejam elas domésticas ou industriais). O que falta, diz ainda Engelberger, é o *software,* isto é, a "inteligência".

1.2. Classificação de Robots

Hoje em dia, e um pouco na linha da separação de objectivos referida no início, existe uma distinção entre os domínios da robótica móvel e da robótica "fixa". São exemplos dos primeiros as plataformas móveis ou ainda os robots com "pernas" desenvolvidos na Universidade de Carnegie Mellon (CMU) e noutros sítios. O outro campo da robótica é dominado essencialmente pelos braços articulados, encontrando-se actualmente numa fase bastante avançada e muito divulgados a nível industrial, nomeadamente no ramo automóvel — são normalmente designados por **robots industriais**. Estes dois grandes campos da robótica, que não definem uma distinção formal na classificação de robots, distinguem-se sobretudo pela forma com que os robots interagem com o ambiente, mais precisamente, nas

questões de percepção e interpretação da informação necessária para levar a cabo uma dada tarefa.

Esta não é de forma nenhuma a classificação mais usada; é tão simplesmente uma separação de categorias baseada num problema fundamental que existe numa categoria e não existe na outra. O primeiro género de classificação que geralmente ocorre é aquele associado à perspectiva histórica: isto é, classificar os robots em gerações. Quatro ou cinco gerações são normalmente enumeradas e distinguem-se sobretudo pelo nível de autonomia e pela capacidade de adaptação a situações novas.

Outra classificação usada é aquela feita pela associação de robótica japonesa (JIRA) que categoriza os robots em 6 classes com base no seu índice de "inteligência". Parte de dispositivos simplesmente manipulados até aquilo a que chamam os robots inteligentes que percebem e interagem com as modificações do ambiente.

Uma outra classificação geralmente adoptada nos laboratórios de investigação, tem a ver com o nível da linguagem de programação usada. Três categorias são normalmente aceites: 1- sistemas guiados, onde o utilizador define quais os movimentos a ser efectuados; 2- sistemas programados ao nível do robot, onde o utilizador escreve um programa que determinará o movimento e a percepção; 3- sistemas de programação ao nível de tarefas ou objectivos, onde o utilizador especifica apenas as operações que o robot deverá efectuar nos objectos e/ou ambiente com que lidará: trata-se de programação ao mais alto nível de abstracção, onde os detalhes para levar a cabo uma dada tarefa são perfeitamente transparentes ao utilizador.

Retornando à classificação baseada em robótica móvel e robótica fixa, no estado actual de desenvolvimento, apresenta-se mais imediata a execução duma acção do tipo "**mudar este objecto deste ponto para aquele ponto [dentro do raio de acção de um braço]**" do que uma do tipo "**ir deste ponto desta sala para aquele ponto desta [ou doutra] sala**". Este facto leva-nos imediatamente a um dos problemas fundamentais da robótica móvel: o problema da localização. Para um braço articulado é sempre conhecido o estado das suas juntas ou das suas extensões telescópicas. À parte as avarias e os desgastes, as interacções mecânicas destes componentes são sempre sabidas graças aos sensores que as guarnecem. Em contrapartida, quando se trata de uma plataforma móvel, é fácil que se verifiquem dessincronizações entre o sistema intrínseco de medição de deslocamentos e os deslocamentos reais: basta pensar nos deslizamentos devido às irregularidades do pavimento, ou aos efeitos de patinagem e derrapagem das rodas. Particularmente sensível a estes efeitos, é sobretudo a orientação do robot. O mais grave nestes erros é o facto de serem cumulativos, e a experiência mostra que ao fim de pouco tempo existe um desfasamento completo entre as coordenadas medidas no sistema do robot e as coordenadas reais medidas por um operador externo. Este método de utilizar exclusivamente a **odometria** do robot (isto é, o sistema associado à

medição intrínseca dos deslocamentos do robot) costuma ser designado com o termo *dead-reckoning*†.

Este problema só poderá ser resolvido recorrendo a outros sensores que prescrutem o ambiente e ajudem o robot a progredir no espaço, e/ou a localizar-se sempre que se encontre num ambiente conhecido.

A navegação é, portanto, o problema principal e o objectivo por excelência das aplicações da robótica móvel. Para a poder fazer, duma forma mais ou menos autónoma, os veículos robóticos têm de dispor de sensores para a percepção do ambiente, e necessariamente de capacidade de interpretação, sendo fundamental realçar esta última, dada por vezes a natureza intrincada, dúbia ou até errónea dos dados sensoriais.

1.3. Navegação de Robots

Navegação é um termo usado quase exclusivamente na robótica móvel, dado que os problemas duais no campo da robótica fixa têm uma nomenclatura quase própria (as cinemáticas), uma vez que aí, a questão é tratada de forma bastante diversa, como já mencionado. Segundo Phillip McKerrow [McKerrow93], navegação é assim definida:

> "Navegação é a ciência (ou a arte) de dirigir o curso de um robot móvel à medida que atravessa o ambiente (terra, mar ou ar)"

Mais concretamente, pode-se afirmar que a navegação é todo um conjunto de componentes que pode incluir elaboração de mapas, planeamento de trajectórias, localização, desvio de obstáculos, saída e recuperação de trajectórias, etc, e que tem geralmente como objectivo levar um robot até um determinado ponto sem se perder e sem embater em nada.

O conhecimento ou desconhecimento *a priori* do ambiente é um factor determinante das acções que se pretendem tomar. Num ambiente desconhecido não é possível fazer planeamento global de trajectórias, e a navegação resulta simplesmente de acções locais (navegação local), e possivelmente também em função do passado. Isso significa que se pode ir acumulando conhecimento sobre o ambiente, podendo como resultado ir construindo um mapa dos locais de passagem. O conhecimento total do ambiente, em contrapartida, viabiliza a maioria das acções de planeamento, carecendo sempre o sistema de um meio de localização para compensar os erros de *dead-reckoning*.

A maioria dos ambientes na prática serão sem dúvida aqueles sobre os quais se tem um conhecimento parcial. Isto é, dispor-se-á de um mapa aproximado duma sala, mas não se sabe se possam ou não existir outros objectos (obstáculos) além dos definidos no mapa inicial. Surge aqui o tema de uma área de larga exploração na robótica: o desvio de obstáculos (*obstacle avoidance*). Assim, supondo um conhecimento contínuo sobre a localização do

† O *dead-reckoning* pode também incluir o recurso a sensores que não a odometria: entre eles os sensores inerciais como acelerómetros, giroscópios, etc. Ou seja, meramente sensores a bordo do robot destinados a medir grandezas (distância, orientação, etc) de forma cumulativa e absoluta.

robot e dado que se sabe também o seu objectivo, desenvolveram-se técnicas de desvio de obstáculos que, de qualquer modo, e em geral, nunca esquecem o ponto de destino final, e orientam todo o procedimento com base nesse facto. Exemplos bem conhecidos podem ser encontrados nos trabalhos de Krogh [Krogh84] e Borenstein [Borenstein89].

A construção do mapa de um ambiente (*environment mapping*), é uma área que tem dado resultados interessantes donde se destacam largamente os trabalhos de Elfes, Moravec [Moravec85] [Elfes89] ou de Zelinsky [Zelinsky88]. O princípio consiste em definir um mapa vazio de um ambiente mediante uma grelha quadriculada mais ou menos fina. Usando dados sensoriais (sobretudo através de ultra-som), e nas sucessivas passagens pelo ambiente, usando modelos probabilísticos para as medições, consegue-se ir criando um mapa grosseiro desse ambiente, frequentemente as salas dos laboratórios. Em complemento ou até alternativamente a esta construção do ambiente por pontos ou pequenas regiões de espaço livre ou ocupado, podem-se tentar extrair indicadores de mais alto nível do ambiente (*features*), tais como segmentos de recta e vértices de obstáculos. Neste técnica são conhecidos, entre outros, os trabalhos de Crowley [Crowley89] e Leonard. Note-se, neste ponto, a referência preferencial a métodos que adoptam sensores do tipo ultra-sónico; os motivos desse facto serão mais claros quando se mencionarem, mais adiante, os sensores mais usuais em robótica móvel. Porém, a construção de mapas pode recorrer, e recorre já actualmente, a outro tipo de sensores (*laser* e vídeo), em especial para a construção de mapas tridimensionais do ambiente como acontece no campo da visão artificial.

1.3.1. O cerne da navegação: os mapas.

O conceito de **mapa** é por vezes usado em navegação com um âmbito mais largo do que simplesmente a planta arquitectónica de uma sala. Um **mapa** pode ser um conjunto de trajectórias possíveis ou a executar, pode ser uma descrição do espaço livre num dado ambiente, ou pode ainda ser uma representação dinâmica do ambiente como é o caso das grelhas de ocupação introduzidas por Elfes. Daqui, se induz de imediato a importância e a necessidade de um mapa. Se não se possui o mapa (neste sentido lato), então geralmente dever-se-á iniciar uma fase de aprendizagem e construir um a partir do ambiente usando os dados sensoriais. Embora posto doutra forma, este acaba por ser ainda o problema principal da robótica móvel: a navegação!

Se inicialmente existir um conhecimento completo do ambiente, as técnicas de planeamento de trajectórias fornecem o mapa sob a forma de listas de trajectórias, que podem ir de percursos pré-determinados até ao traçado óptimo de trajectos que maximizam as distâncias de todos os pontos de passagem a todos os pontos do ambiente, como é o caso dos **diagramas de Voronoi**.

O planeamento de trajectórias (*path planning*) é, só por si, um campo extraordinariamente vasto e conta com inúmeros trabalhos como veremos mais à frente. Tal como o nome o pode sugerir, para planear trajectórias é necessário o conhecimento *a priori* do

ambiente: quais são os limites e as dimensões dos objectos presentes no espaço de trabalho. As técnicas de planeamento partem essencialmente desde os trabalhos de Lozano-Perez sobre o **espaço de configuração.** Estas técnicas, por vezes, fornecem apenas traçados poligonais que podem ser posteriormente suavizados e convertidos em linhas de curvatura mais adequada à navegação.

Descrever-se-ão a seguir os principais trabalhos sobre as diversas componentes da navegação de robots móveis. Os tópicos são os seguintes:

1) Planeamento de trajectórias
2) Desvio de obstáculos
3) Métodos de localização
4) A construção de mapas de ambientes

1.3.2. Planeamento de trajectórias

A área do planeamento de trajectórias e desvio de obstáculos é conhecida em geral pela expressão **Planeamento do Movimento** (*Robot Motion Planning*). É usual dividir as técnicas de planeamento do movimento em duas categorias: técnicas baseadas no **Espaço de Configuração,** e técnicas baseadas em **Campos de Potenciais Artificiais.** Contudo, esta separação nem sempre é vista como legítima dado também se apoiar que as técnicas baseadas em campos de potenciais se derivam da teoria do espaço de configuração [Latombe91]. Contudo, a técnica de campos de potenciais, pelo interesse que tem suscitado nos últimos anos e pela forma com que aborda o problema do planeamento do movimento, neste texto será tratada na secção do desvio de obstáculos, cuja componente de "planeamento" é de cariz local e não global como o planeamento de trajectórias.

Espaço de configuração é uma definição da geometria computacional popularizada por Lozano-Perez a partir sobretudo dos seus trabalhos sobre **Planeamento Espacial** (*Spatial Planning*) [LozanoPerez83]. Nesses trabalhos propôs algoritmos para a determinação do conjunto de pontos de espaço livre onde poder colocar um objecto (robot) num espaço contendo corpos poligonais convexos (obstáculos) — tratava-se do *Findspace Algorithm*. De igual modo, propôs o *Findpath Algorithm* onde se calculavam as trajectórias para ir dum ponto a outro daquele mesmo espaço, evitando os obstáculos nele contidos: este algoritmo tomava como base o algoritmo anterior (*Findspace*).

As técnicas baseadas no espaço de configuração requerem um conhecimento completo da lista de obstáculos no ambiente. Para a definição de uma trajectória requerer-se-á ainda, como é de esperar, o início e o fim dessa trajectória, ou seja, o ponto de partida e o objectivo final para onde se pretende mover.

O grande número de métodos para fazer o planeamento da trajectória de um robot divide-se em geral em duas grandes categorias: "mapa de caminhos" (*roadmap*) e decomposição em células (*cell decomposition*).

1.3.2.1. Mapa de caminhos (*Roadmap*)

Nesta categoria de métodos, o princípio fundamental é o de extrair uma rede de trajectos unidimensionais (curvas) através do espaço livre entre os obstáculos: o "mapa de caminhos". Os métodos mais conhecidos são os **Grafos de Visibilidade** (*Visibility graphs*) e os métodos de **retracção** (*retraction*) que num espaço a duas dimensões geram os conhecidos **diagramas de Voronoi.**

Os grafos de visibilidade [Nilsson69] são um dos métodos mais antigos para o traçado de trajectórias e geram percursos semi-livres (*semi-free paths*) percorrendo os segmentos que unem em linha de vista os vértices dos obstáculos poligonais do ambiente e os pontos inicial e final da trajectória (fig. 1). A expressão **percurso semi-livre** está associada ao facto que as trajectórias "tocam" os vértices dos obstáculos, carecendo deste modo de ajustes posteriores para uma implementação prática útil.

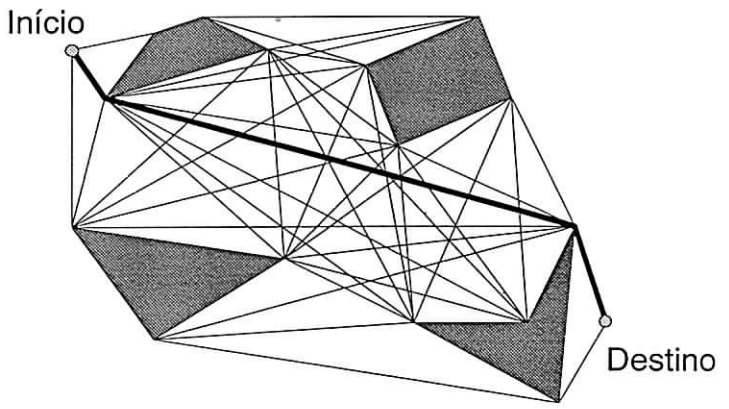


Fig. 1 - Grafo de visibilidade. O caminho mais curto é calculado seguindo os segmentos que unem todos os vértices mutuamente visíveis dos obstáculos poligonais e os pontos de partida e chegada.

Existem múltiplas implementações do algoritmo para construir o grafo de visibilidade cujos custos de computação podem variar desde $O(n^2 \log n)$ até valores mais recentes como $O(n^2)$ [Latombe91], sendo n o número de vértices dos obstáculos do espaço. O algoritmo conclui-se com a procura do trajecto mais curto determinando a sequência de segmentos adequada.

O princípio subjacente ao método de retracção está ligado ao conceito original em topologia, mas que pode ser sintetizado do seguinte modo: retracção é a transformação de um qualquer espaço E num sub-espaço F, e que é igual à transformação identidade quando aplicada a F [Sharir89]. Isto significa que a transformação sucessiva de um espaço resultará, o mais tardar a partir da segunda iteração, no próprio espaço (idempotência). O resultado de uma retracção dum espaço bidimensional é designado por **Diagrama de Voronoi** [O'Dunlaing85], que é um conjunto de linhas contínuas e ligadas entre si (conexas) cuja característica principal de todos os seus pontos é encontrarem-se à máxima distância dos obstáculos (objectos) do espaço inicial. Uma analogia na análise e processamento de imagem são os algoritmos de esqueletização. Um Diagrama de Voronoi generalizado é o lugar

geométrico de todos os pontos equidistantes de dois ou mais obstáculos incluindo os limites do espaço de trabalho (fig. 2).

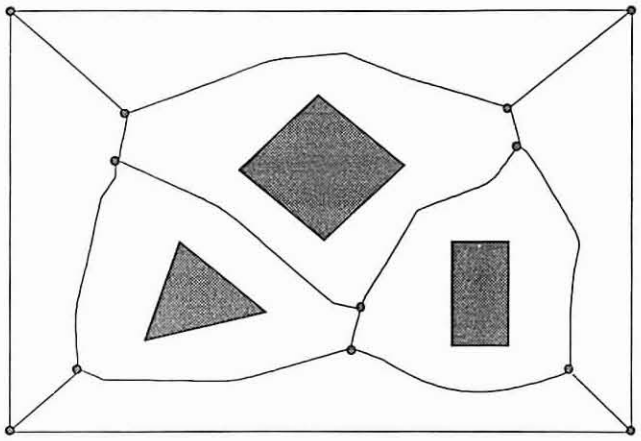


Fig. 2 - Esquema de um Diagrama de Voronoi Generalizado: trajectos óptimos em termos da distância máxima aos obstáculos e paredes do ambiente.

A execução de uma trajectória seguindo um diagrama de Voronoi generalizado requer ainda alguma heurística no movimento para tomar em conta as contingências geométricas do robot [Takahashi89], ou seja as suas dimensões e/ou graus de liberdade. O custo computacional dos diversos algoritmos para calcular os diagramas de Voronoi vai desde o pesado e *naïf* $O(n^4)$, passando por $O(n \log^2 n)$ até ao mais eficiente $O(n \log n)$, sendo n o número total de vértices dos obstáculos [Latombe91].

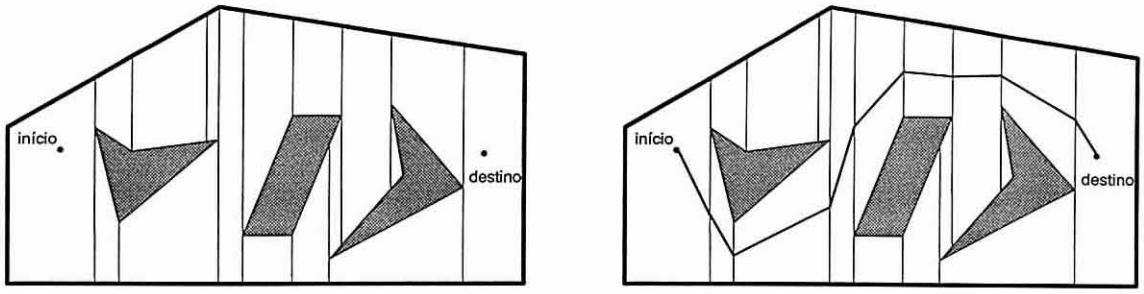
Outros métodos desta categoria de "Mapa de caminhos", à parte o método da **silhueta** (*silhouette*) [Canny88], acabam por ser extensões ou adaptações dos métodos descritos. O método da silhueta distingue-se pela sua capacidade de não se limitar a espaços de apenas duas ou três dimensões. De facto, qualquer problema que possa ser reduzido a um planeamento de trajectória pode ser resolvido com este método; isso inclui, por exemplo, múltiplos robots partilhando o mesmo espaço. O algoritmo tem funcionamento recursivo e é de implementação complexa, sendo também por isso menos frequente nas aplicações de planeamento em robótica móvel (espaços a duas dimensões), onde os algoritmos descritos anteriormente são em geral satisfatórios.

1.3.2.2. Decomposição em células (*Cell decomposition*)

Os métodos de decomposição em células assentam numa técnica que consiste em segmentar todo o espaço livre em regiões adjacentes, traçar um **grafo de conectividade** (*connectivity graph*) que relaciona a propriedade de adjacência de regiões, determinar um "canal" que inclua os pontos de partida e chegada e finalmente extrair uma trajectória seguindo um critério adequado dentro do "canal" determinado.

O conceito de partida para este tipo de métodos é o de **decomposição poligonal convexa**, que se relaciona com a fragmentação de uma região do plano (o espaço livre) em

polígonos convexos não intersectados e adjacentes. Uma ilustração desta metodologia é a decomposição trapezoidal mostrada da sequência na figura 3.



(a) (b)

Fig. 3 - a) Decomposição trapezoidal do espaço livre. b) Traçado de uma trajectória pelos pontos médios dos segmentos do "canal" encontrado pelos algoritmos.

A decomposição em células divide-se em duas subcategorias: **decomposição exacta** e **decomposição aproximada**. A segunda distingue-se da primeira pela necessidade da forma rectangular das células (regiões) em que se divide o espaço livre. Com uma representação assim é impossível manter uma definição exacta dos obstáculos e do ambiente, daí o nome de decomposição aproximada. A decomposição exacta foi inicialmente desenvolvida por Schwartz e Sharir [Schwartz83] que propuseram de igual modo uma variante do algoritmo, por sinal bastante ineficiente segundo os próprios autores, que resolve o caso geral do planeamento de trajectórias.

A decomposição aproximada do espaço é uma técnica que produz células (regiões), geralmente quadradas, e de três tipos conforme se situem completamente em zonas de espaço livre, em zonas de obstáculo ou ainda em zonas mistas de espaço livre e obstáculo, ou seja nas fronteiras dos obstáculos. A decomposição aproximada foi primeiramente introduzida por Lozano-Pérez e Brooks [LozanoPerez81][Brooks83].

1.3.3. Desvio de obstáculos

Na grande maioria dos métodos descritos anteriormente nunca há uma distinção clara entre o que é o planeamento de uma trajectória e o desvio de obstáculos. As duas funções estão automaticamente ligadas quando se planeia o percurso a executar para levar a cabo uma dada tarefa: assume-se que a trajectória é cumprida tal como foi planeada. Como esta ideia é muito irrealista quando se fala de robótica móvel, surgiu de imediato a necessidade de conceber um **planeamento de trajectórias adaptativo ou exploratório** [Sharir89], ou ainda a expressão **desvio de obstáculos em tempo real** como por exemplo em [Borenstein89], para distinguir do desvio de obstáculos inerente ao planeamento global de uma trajectória.

É imperativo, antes de mais, clarificar as definições de **ambiente** e de **obstáculo**. É mais ou menos intuitivo que a distinção assenta sobre o que é conhecido e desconhecido num dado espaço ou meio: **ambiente** é tudo aquilo que, *a priori*, se conhece e pode descrever no que

respeita a dimensões físicas e localização geográfica. Isso inclui todos os objectos no meio de navegação, como paredes, mobília ou outros elementos cuja ocupação espacial seja relevante para as tarefas de navegação. **Obstáculo** será tudo o que é desconhecido à partida e que, portanto, não é levado em conta para planear qualquer trajectória no meio em causa. Estas definições são algo empíricas e a pertinência do seu uso ocorre precisamente quando se pretende distinguir planeamento de trajectória e desvio de obstáculos. É assim claro que o **planeamento de trajectórias** não encara os obstáculos como tal; sendo conhecidos, são parte do ambiente, logo levados em conta. O **desvio de obstáculos** baseia-se na detecção do que não pertence ao ambiente conhecido e consiste em tomar as decisões apropriadas para a continuação da navegação, procurando variantes locais às trajectórias planeadas evitando esses elementos desconhecidos que podem pôr em risco a segurança do robot.

O desvio de obstáculos foi tentado por múltiplos autores usando métodos baseados na detecção das arestas "verticais" dos obstáculos, como por exemplo em [Borenstein88] ou [Crowley89]. Contudo, as dificuldades associadas a esta detecção, nomeadamente pelas limitações dos sensores e pela necessidade de fazer uma percepção do ambiente em posição estática, remeteram esta técnica para um plano secundário, especialmente quando métodos como os campos de potenciais oferecem possibilidades de navegação mais versáteis e promissoras, como veremos de seguida.

Em oposição ao planeamento de trajectórias, os métodos de desvio de obstáculos (em tempo real) são também denominados métodos de cariz **local.**

1.3.3.1. Campos de potenciais artificiais (*Artificial Potential Fields*)

O método dos campos de potenciais artificiais assentam no conceito de um campo virtual de forças atractivas exercidas pelo ponto de destino, e forças repulsivas exercidas pelos obstáculos. A grandeza destas forças depende da distância do robot a esses pontos criadores das forças. Define-se assim uma função de potencial mais ou menos complexa dependente do número de obstáculos no espaço de configuração.

Esta ideia foi desenvolvida por Kathib [Kathib85] e Krogh [Krogh84], e foi depois alargada também por Krogh e Thorpe [Krogh86] para uma solução integrada de planeamento de trajectórias e desvio de obstáculos em tempo real.

Sendo uma técnica caracterizada por gerar uma grandeza gradiente, corre o risco de produzir soluções (ponto de chegada) em mínimos locais, em vez do mínimo absoluto que seria o ponto de destino. Mas, definir uma função potencial que possua um único mínimo (coincidente com o local de destino) equivale a tirar o carácter **local** desta técnica, uma vez que se teria de entrar em conta com todos os pontos do ambiente, ou seja o conhecimento **global** de todos os obstáculos e suas posições.

$$U = U_{att} + U_{rep} \qquad (1)$$

A função potencial U é dada pela soma dos potenciais atractores (U_{att}) e dos potenciais repulsores (U_{rep}) como em (1). Em geral, o foco de atracção é um só (o ponto de destino), mas os pontos de repulsão são tantos quantos os obstáculos (pontuais) existentes no espaço de configuração, resultando num potencial de repulsão dado pela expressão (2).

$$U_{rep} = \sum_{i=1}^{N} U_{rep_i} \qquad (2)$$

sendo N o número de pontos repulsores no ambiente

A função para definir o potencial de atracção pode seguir uma simples lei quadrática com a distância Euclidiana [Kathib85] à semelhança das funções potencial das leis da Física e, tal como essas, tem um raio de acção infinito, isto é, afecta todos os pontos do espaço (Eq. 3).

$$U_{att}(\mathbf{d}) = \frac{1}{2}\xi\rho^2_{goal}(\mathbf{d}) \qquad (3)$$

sendo $\mathbf{d}$ *o vector distância ao ponto atractor,*
ξ *uma constante positiva e*
ρ_{goal} *a função de cálculo da distância ao ponto de destino.*

A função para o potencial de repulsão (vinda dos obstáculos) obedece geralmente a um preceito fundamental: o robot nunca deverá ser capaz de atravessar um obstáculo, ou seja o potencial repulsor tende para infinito à medida que aumenta a proximidade desse ponto (Eq. 4).

$$U_{rep_i}(\mathbf{d}_i) = \frac{1}{2}\eta\left(\frac{1}{\rho(\mathbf{d}_i)}\right)^2 \qquad (4)$$

sendo $\mathbf{d}_i$ *o vector distância a cada ponto repulsor,*
η *uma constante positiva e ,*
ρ *a função de cálculo da distância a cada ponto.*

Além da condição anterior sobre a intransponibilidade dos obstáculos, pode ser também desejável que um obstáculo não afecte o movimento do robot quando este passa a uma distância superior a um determinado limite desses obstáculos (Eq. 5).

$$U_{rep_i}(\mathbf{d}_i) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{\rho(\mathbf{d}_i)} - \frac{1}{\rho_0}\right)^2 & \Leftarrow \rho(\mathbf{d}_i) \le \rho_0 \\ 0 & \Leftarrow \rho(\mathbf{d}_i) > \rho_0 \end{cases} \qquad (5)$$

sendo $\mathbf{d}_i$ *o vector distância a cada ponto repulsor,*
η *uma constante positiva,*
ρ *a função de cálculo da distância a cada ponto e,*
ρ_0 *a distância limite de influência de um obstáculo*

Como exemplo numérico podemos considerar uma situação em que se definem dois obstáculos pontuais num ambiente rectangular e (como é normal) um ponto atractor que é o ponto de destino (fig. 4).

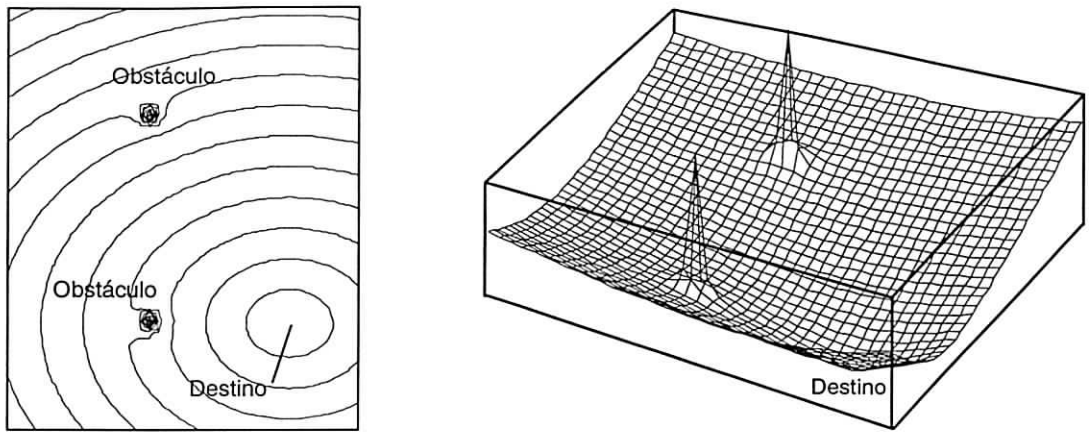


Fig. 4 - Exemplo de um campo de potenciais com dois pontos repulsores (obstáculos pontuais) e um ponto atractor (destino). a) Contornos equipotenciais. b) Representação tridimensional da função potencial no plano do espaço de configuração

Uma vez calculada a função potencial em cada ponto do espaço, a determinação da força resultante de atracção para o ponto de destino indicará a nova direcção de movimento e a sua intensidade. A força é calculada mediante a seguinte expressão (6):

$$\vec{F}(\mathbf{d}) = -\vec{\nabla}U(\mathbf{d}) \qquad (6)$$

A trajectória resultante será sempre normal às curvas equipotenciais no campo do espaço de configuração (fig. 5). Trata-se tão simplesmente da característica normal de uma grandeza gradiente (força) definida num campo de escalares (potenciais).

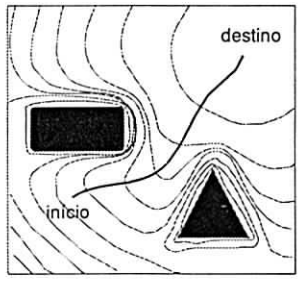


Fig. 5 - Curvas equipotencial para um caso com 2 obstáculos não pontuais. Notar como a trajectória resultante intersecta ortogonalmente as curvas equipotenciais

1.3.3.2. Derivações e complementos ao método dos campos de potenciais

Como já mencionado, o método dos campos de potenciais corre o risco de gerar uma trajectória até um mínimo local, ou seja a força resultante é nula (fig. 6). Surge assim a necessidade de introduzir um qualquer tipo de heurística para evitar estas configurações indesejadas: o princípio usualmente sugerido é o de prosseguir a trajectória seguindo os contornos dos obstáculos [Donald87].

Variantes do método dos campos de potenciais descrito incluem, por exemplo, os já referidos trabalhos de Krogh sobre os **campos de potenciais generalizados** [Krogh84], onde as forças de repulsão actuam apenas se houver a tal proximidade do obstáculo em causa, e ainda apenas se o robot se estiver a deslocar na direcção desse obstáculo, como reforça também Tilove [Tilove90]. Em consequência, se o robot se estiver a mover paralelamente a um dado obstáculo, então não será repelido por este último.

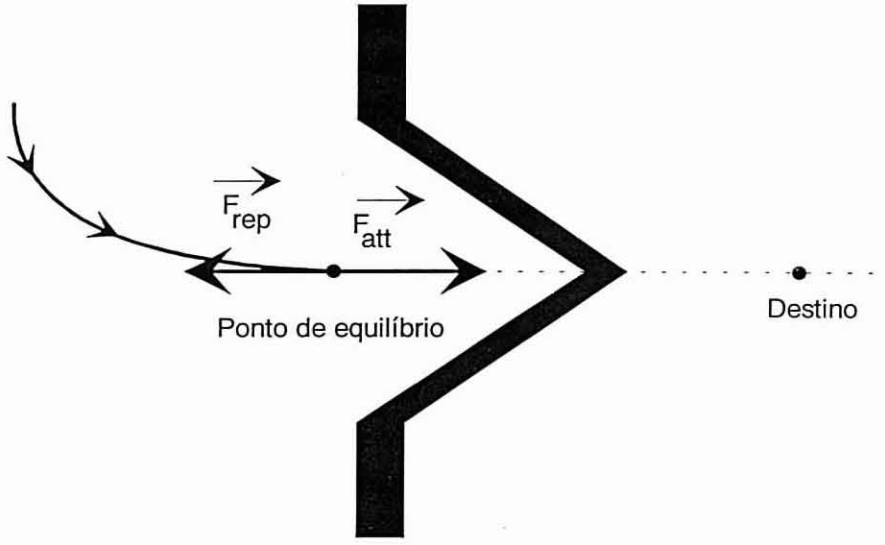


Fig. 6 - Exemplo de um mínimo local numa trajectória gerada pelo método dos campos de potenciais.

A construção analítica de uma função de navegação para espaços de configuração com geometrias arbitrárias pode ser bastante complexa, donde se opta em muitos casos por uma solução baseada na representação discretizada (formato de grelha) do espaço de configuração [Barraquand91]. Esta solução é independente da geometria do espaço de configuração, mas torna-se computacionalmente muito complexa quando se abordam problemas a mais de 2 ou 3 dimensões geométricas, o que no entanto é quase irrelevante para a prática.

Borenstein tinha ainda proposto em 1989 [Borenstein89] o **campo de forças virtuais** (*Virtual Force Field*) como uma combinação de campos de potenciais para a navegação, e de **grelhas de certeza** (*Certainty grids*) para a representação aproximada dos obstáculos no espaço de configuração. Este método, salienta Borenstein, é adequado à representação aproximada do ambiente que normalmente se obtém dos sensores, nomeadamente pelo carácter incerto ou probabilístico dos dados sensoriais.

A dinâmica da percepção e da navegação é um factor de extrema importância. Num contexto de navegação local (distinto de planeamento global de trajectórias) a técnica "pára-olha-e-segue" deve ser substituída por algo mais dinâmico e eficiente. Esta propriedade da "continuidade" da navegação é atingida (pelo menos em simulação) com a definição dos **sub-objectivos** (*subgoals*) na execução de determinadas trajectórias nos trabalhos de Krogh e Feng [Krogh89][Feng90]. Nos algoritmos propostos, este autores definem funções de controle da direcção de movimento do robot através de dados captados por sensores.

Dentro do campo do desvio de obstáculos uma última referência importante são os trabalhos recentes de Borenstein [Borenstein91] sobre o **histograma de campo vectorial** (*Vector Field Histogram*). Este método, que deriva das mesmas técnicas dos campos de potenciais, é uma versão refinada dos trabalhos anteriores deste autor sobre o **campo de forças virtuais**. A novidade introduzida relaciona-se com a representação do ambiente que rodeia o robot (espaço local) em forma de histograma polar, e que tenta levar em conta a natureza dos sensores, em especial o ruído de medição que caracteriza os sensores de ultra-som, mas não conta com a redundância espacial dos sensores, apenas com estatísticas temporais das medições de cada sensor. O método consiste em seleccionar os "corredores" (*valleys*) de espaço desimpedido que mais se aproximam da direcção do ponto de destino. Existe também aqui a referência ao objectivo final da navegação.

1.3.4. A localização

A localização é por excelência um conceito inerente à robótica móvel. Por localização entende-se o conjunto de procedimentos que permitem ao robot determinar a sua posição (local geográfico) num qualquer referencial fixo ao ambiente. Um processo explícito de localização é necessário devido aos erros acumulados pelo sistema de *dead-reckoning*. Como é lógico, um robot não poderá executar correctamente uma trajectória pré-calculada se não tiver conhecimento da sua posição corrente.

Em geral, o sistema de **odometria** fornece uma primeira estimativa da posição que é depois utilizada para um cálculo mais preciso da localização do robot. Mas nem sempre é assim: Drumheller propôs a **localização absoluta** [Drumheller87] em que sustentava a capacidade de localização independentemente da história do movimento ou de quaisquer indicadores iniciais de posição.

A localização com o auxílio de referências conhecidas no próprio ambiente é um processo muito eficiente e assenta frequentemente em métodos de triangulação. Estes métodos consistem em medir a distância a vários pontos conhecidos e identificáveis no ambiente para uma dada posição do robot, ou medir a distância a um desses pontos conhecidos de várias posições do veículo. A primeira opção é preferível à segunda, mas se o número disponível desses pontos do ambiente é muito escasso (só um, por exemplo) os cálculos geométricos para a triangulação exigem posições suplementares do robot. A desvantagem é que se corre o risco de se introduzir erro, mesmo que pequeno, nestas deslocações do veículo.

As referências no ambiente podem ser activas ou passivas. No primeiro caso temos os conhecidos sistemas de **faróis** (*beacons*) que podem ser emissores de rádio ou infravermelho, ou até de ultra-som como por exemplo em [Kleeman92]. O robot leva a bordo os receptores adequados e contará com a distribuição dos emissores pelo ambiente para calcular de forma contínua a sua posição. As referências passivas são em geral marcas especiais como figuras geométricas simples e bem visíveis (*landmarks*) com grande multiplicidade de formatos

[Feng92] e/ou códigos de barras que serão interpretados por sistemas de captação de imagem ou ainda leitores laser.

A desvantagem de sistemas como estes é que obrigam a uma "invasão" do ambiente para a colocação prévia destas marcas ou referências. Não sendo isso muitas vezes possível nem desejável, e até para tornar um sistema de localização mais independente do ambiente, tem-se procurado efectuar a localização com base no próprio ambiente ou em detalhes particulares desse ambiente.

A ideia original de Drumheller [Drumheller87] foi a de usar um grande número de leituras ultra-som em torno do robot e calcular a posição na sala que melhor se adaptasse a esse conjunto de leituras. A qualidade dos resultados era encorajadora, mas a principal conclusão é que melhores resultados seriam possíveis caso se usassem sensores doutra natureza que não ultra-som: um dos principais problemas era a resolução angular das medidas. A localização através de ultra-som num ambiente conhecido, mas que inclui possivelmente obstáculos desconhecidos, foi ainda estudado por outros autores como por exemplo em [Holenstein92].

Um dos trabalhos mais importantes na localização baseada exclusivamente no mapa do ambiente foi desenvolvido por Cox [Cox91]. O sistema usa a **odometria** do seu robot, Blanche, e um leitor óptico de distâncias. Este leitor óptico faz uma sequência de leituras em torno do veículo, e o conjunto de pontos de medição resultante é comparado com uma representação do mapa da sala (segmentos de recta). O algoritmo ajusta a sobreposição dos pontos e do mapa de forma a minimizar as distâncias entre eles (fig. 7).

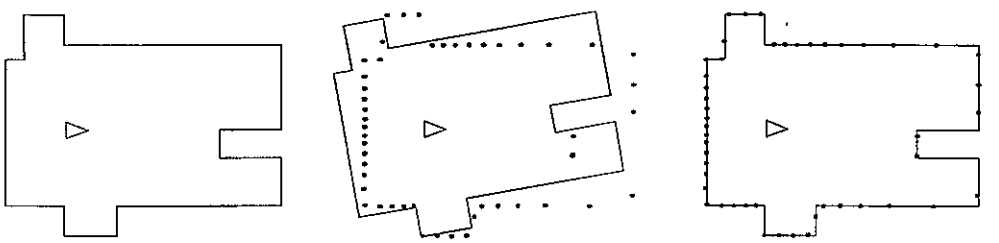


Fig. 7 - Sistema de localização baseado no alinhamento de medidas sensoriais e num mapa *a priori*.
a) Posição real do robot. b) Localização esperada pelo robot e pontos de medição ao ambiente.
c) Determinação da posição correcta por alinhamento das medidas e do mapa conhecido.

Um outro trabalho importante na área da localização foi desenvolvido fundamentalmente por Leonard e Durran-Whyte [Leonard92] incluindo algumas variantes como por exemplo em [Pedrosa94b]. Ao contrário de Cox, estes investigadores, tal como Drumheller e outros, usaram dados de ultra-som para efectuar a localização. O princípio foi o de usar **Filtros de Kalman** como uma ferramenta estimadora da posição corrente em função das posições anteriores e das leituras sensoriais. A representação do ambiente é também de natureza geométrica (linhas, pontos,....), e não usa o formato em grelha que corresponderia a uma discretização do espaço em células livres e ocupadas.

Uma retrospectiva sobre os métodos de localização pode ser encontrada num trabalho bastante exaustivo de L. Feng, J Borenstein e H. Everett [Feng94]. Este trabalho é igualmente uma referência importante no que respeita aos sensores existentes para permitir a localização.

1.3.5. A construção de mapas de ambientes

Construir um mapa de ambiente pode ser muito importante em certas situações. Quando um ambiente é desconhecido e se pretende fazer planeamento de trajectórias nesse ambiente (para outras missões, por exemplo), a questão principal é construir o mapa antes de qualquer outra acção. A construção de mapas pode ser dividida em duas categorias [Leonard90]: determinação do **modelo geométrico** e **grelhas de ocupação.**

O método da determinação do modelo geométrico consiste em extrair as linhas (contornos) dos objectos e criar assim uma representação compacta e realista do ambiente. Em geral, estes métodos estão ligados a trabalhos de visão (imagem), em particular com estereoscopia binocular [Kriegman89] e triocular [Ayache89]. Contudo, esta procura de uma representação do mapa através de descritores geométricos foi também experimentada por Crowley [Crowley89] usando ultra-som.

A técnica de **grelhas de ocupação** consiste em criar uma representação discretizada do ambiente em forma de grelha quadriculada, cujas células têm associado um valor que é a ocupação do espaço na região correspondente. Esta técnica é particularmente interessante quando os dados sensoriais são imprecisos e porque torna possível a integração (fusão) de múltiplos tipos de sensores (por exemplo, infravermelho e sonar). Os trabalhos mais conhecidos são os de Moravec e Elfes [Moravec85] [Elfes89]. Nesses métodos, cada célula tem associada uma probabilidade de ocupação que é actualizada sucessivamente de cada vez que são feitas as leituras sensoriais. As múltiplas passagens e mudanças de posição do robot modificam o ponto de vista do ambiente e assim os valores das células são refinados com o decorrer do tempo. O método prevê a monitorização da localização para compensar os erros de *dead-reckoning*.

1.4. Sensores

Os sensores (também designados por sondas) são os componentes que permitirão uma "percepção" do mundo exterior. Esta percepção é feita através da medição ou avaliação duma variedade de grandezas físicas que podem ir da simples medição de distâncias à interpretação relativa dos níveis de iluminação ou de cor numa ou mais imagens de vídeo ou de infravermelhos. A percepção propriamente dita, isto é, uma interpretação e análise dos dados sensoriais é em geral feita por um sistema a um nível superior ao dos sensores e que permite uma representação integrada da grandeza ou propriedade que se está a avaliar.

1.4.1. Tipos de sensores usados em navegação

Em navegação, os sensores mais imediatos e mais fáceis de usar são sem dúvida aqueles que permitem medir a distância do robot ao ambiente. São também aqueles que permitirão determinar mais directamente a grandeza de maior importância para a navegação: o espaço livre até ao obstáculo mais próximo!

Dentro dos sensores indicados para a avaliação de distâncias, podem enumerar-se os seguintes como os mais importantes:

- Sensores ultra-sónicos
- Sensores de proximidade/distância com base em infravermelhos
- Sensores de medição laser
- Sensores de contacto (antenas e *bumpers*)

Todos eles, além de se basearem em princípios físicos diferentes, apresentam campos de aplicação distintos com vantagens e desvantagens, uns relativamente aos outros. Os sensores de contacto são de aplicação tão específica, que colocá-los na categoria de avaliadores de distância surge algo forçado. Destinam-se essencialmente a situações extremas, isto é, para as eminências de embate ou mesmo para minimizar os danos em caso de embate.

Os sensores existentes baseados em infravermelhos são usados para percepção de pequenas distâncias (até cerca 1 metro). As primeiras versões deste tipo de sensores geravam informação de natureza binária: existe um obstáculo a uma distância inferior a um dado limite (especificação do sensor) ou não existe obstáculo algum! Não eram desta forma verdadeiros medidores de distância, mas sim, também estes, detectores de proximidade. As últimas evoluções introduziram a medição de distâncias, mas sempre dentro de um restrito campo de distâncias a medir, e o ruído devido à componente infravermelho da luz ambiente (luz solar, sobretudo) dificulta o seu uso. A propriedade reflectora dos materiais (cor, brilho) é de vital importância, pois a distância medida é inversamente proporcional à intensidade do eco: distâncias maiores resultam em ecos menores, logo mais difíceis de distinguir do ruído. Este princípio é válido, nas respectivas extensões como se verá, para outros tipos de sensores.

Da lista anterior restam os sensores à base de ultra-som e de laser *(Laser Range Finders - LRF)*. Estes tipos de sensores fornecem informação, mais ou menos precisa, sobre as distâncias que separam até ao obstáculo alinhado com o feixe de medição, baseando-se frequentemente quer no tempo de voo (*Time of flight - TOF*) quer no desvio de fase (*Phase Shift*). Contudo, este é o único ponto em comum entre estes dois tipos de sensor. Os princípios físicos, as precisões, as velocidades de operação, a dependência com o ambiente e os custos comerciais, entre outros, separam completamente estes dois tipos de sensores.

Uma última categoria, que neste momento não parece ter sido explorada (pelo menos, por aparentes motivos de custo comercial) é a classe dos radares. O princípio de medição baseado no tempo de voo manter-se-ia, usar-se-iam apenas micro-ondas como forma de radiação.

1.4.2. Os sensores ultra-sónicos

A medição de distâncias nas aplicações em robótica está maioritariamente apoiada em sensores de ultra-som. Isto é devido essencialmente ao baixo custo quando comparado com medidores de distância à base de laser. Outras razões para isso incluem: i) a tecnologia de ultra-som é conhecida e dominada há bastante tempo, ii) não apresenta riscos de segurança contra os seres humanos, iii) em circunstâncias adequadas a informação proveniente deste tipo de sensores é bastante precisa, iv) e são relativamente fáceis de controlar.

Dada a sua vasta difusão na comunidade robótica, é pertinente destacar algumas das propriedades, vantagens e problemas dos sensores de ultra-som.

O princípio físico em que se baseiam consiste em medir o tempo que decorre entre a emissão de uma onda acústica de alta frequência e a recepção de um eco devido à reflexão em corpos na trajectória do feixe incidente. Sabendo a velocidade de propagação do som, é imediato deduzir qual a distância mais curta ao ambiente reflector na direcção do feixe emitido.

Esta descrição, por muito sucinta que pareça, contém de certa forma quase toda a problemática da medição de distâncias com ultra-som. Sem dúvida, a expressão mais polémica é "...corpos na trajectória do feixe incidente...". Teoricamente deduz-se com alguma facilidade [Kinsler62], e a prática o demonstra com evidência [Santos93b], que é impossível localizar com precisão um corpo detectado por um feixe de ultra-som usando os sensores comuns. Isto deve-se ao facto de o feixe constituir uma frente de onda esférica, aumentado com a distância a área propícia à reflexão. Deste modo, é possível localizar um corpo quanto à sua componente radial, mas apenas estimar os limites do seu enquadramento angular (Fig. 8).

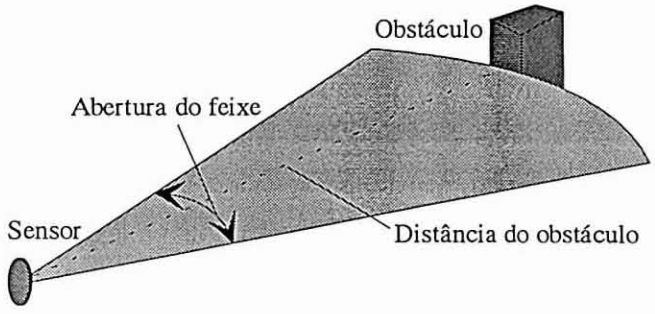


Fig. 8 - A propriedade cónica de um feixe de ultra-som limita a localização angular de um obstáculo.

Esta limitação tem ainda outra consequência: não permite concluir quanto às dimensões do objecto reflector. Isto acontece pelo menos na prática, porque em teoria poder-se-ia esperar avaliar a energia recebida da reflexão supondo ainda que se soubessem quais os coeficientes de absorção dos corpos reflectores. Todavia, isto não é viável em primeira aproximação: muito simplesmente os sensores recebem ecos complexos e geralmente o circuito que os controla baseia-se num critério de energia mínima cumulativa (a emissão é frequentemente feita por meio de uma série de impulsos) para assinalar uma detecção.

O problema da localização angular do obstáculo e/ou das suas dimensões é só uma faceta da questão do "alinhamento com a trajectória do feixe": na verdade, o ângulo de incidência revelar-se-á como a componente mais sensível em todo o acto de medição de distâncias com ultra-som, como veremos de seguida.

Sendo uma onda, a propagação de ultra-som (e do som em geral como uma transmissão de energia através dos materiais) goza de todas a propriedades dos fenómenos ondulatórios. A mais importante no acto de medição de distâncias, e já começada a ser tratada atrás, é efectivamente a reflexão. Dado o comprimento de onda da propagação ultra-sónica mais usada na prática (50 kHz) ser relativamente elevado (~6.9 mm), são notórios à escala macroscópica os efeitos das chamadas **reflexões difusas** e **reflexões especulares**.

As reflexões difusas ocorrem quando os corpos reflectores são de dimensões inferiores ao comprimento de onda do feixe. Nestas condições haverá a garantia da reflexão possuir uma forte componente no sentido oposto à incidência, assegurando a possibilidade de detecção junto ao emissor. Este é o caso de objectos com determinadas texturas ou rugosidades.

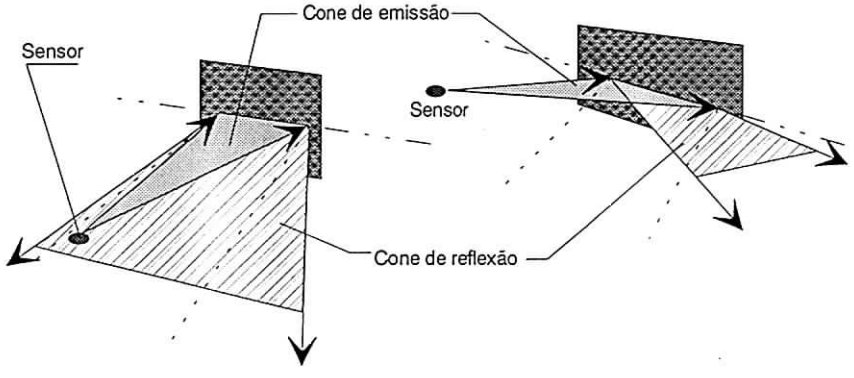


Fig. 9 - Ilustração de como uma reflexão especular pode dificultar a medição com ultra-som.

As reflexões especulares (do latim *speculum*, espelho) ocorrem quando as dimensões dos corpos reflectores ultrapassam o comprimento de onda do feixe incidente. Neste caso aplica-se a lei tão bem conhecida da óptica: o ângulo de reflexão é igual ao ângulo de incidência. Ou seja, corre-se o risco de o feixe reflectido não mais encontrar o sensor (fig. 9). Este efeito da reflexão especular cria uma dependência de tal modo grande com o ambiente que é difícil modelizar um sensor de ultra-som e as medidas que dele se esperam, mormente no caso em que se desconhece o ambiente e/ou a localização do veículo, logo a orientação de um feixe em relação às superfícies dos obstáculos.

Para se poder quantificar estes fenómenos de reflexões especulares não detectadas e de limites de detecção de obstáculos, deve-se analisar a distribuição angular de energia do feixe emitido bem como a sensibilidade do receptor e absorção do feixe à medida que atravessa o ar.

O diagrama de radiação teórico para um sensor de ultra-som do tipo mais usado em robótica, encontra-se deduzido por exemplo nos trabalhos de Kinsler e apresenta-se na

figura 10. Como se pode ver nessa figura, existe energia sónica propagada numa grande região de espaço (em teoria, todo o espaço delimitado pelo plano do sensor contém energia). Contudo, através da equação do diagrama de radiação (Eq. 7) verifica-se de imediato que a potência de emissão decai para metade (-3dB) a um azimute de cerca de 5.4°, definindo assim uma abertura de cerca de 11° dentro da qual a energia emitida será eventualmente suficiente para, após uma reflexão favorável, provocar uma detecção.

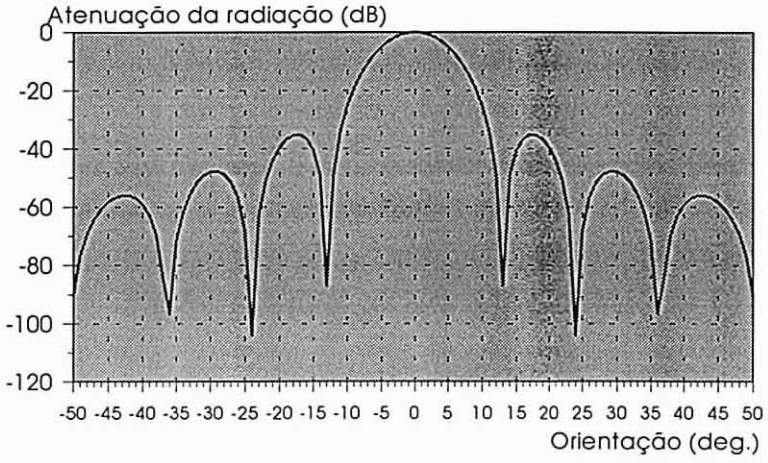


Fig. 10 - Diagrama de radiação para um sensor circular de raio *a* no comprimento de onda λ.

$$P(\theta) = \frac{2J_1(ka\sin(\theta))}{ka\sin(\theta)} \qquad (7)$$

onde k=(2π/λ), a é o raio do transdutor
λé o comprimento de onda do som
$J_1()$ é a função de Bessel da primeira espécie de 1ª ordem.

Porém, e como seria de esperar, a sensibilidade do detector é determinante quanto ao ângulo dentro do qual uma reflexão pode ainda ser detectada. Uma grande sensibilidade fará eventualmente sofrer os efeitos do ruído de reflexões secundárias, mas permitirá a detecção de reflexões com grandes ângulos de incidência. Ao contrário, uma menor sensibilidade do receptor permitirá detecções em estreitas faixas de ângulos de emissão (por exemplo 10°). De qualquer forma, na prática o ganho do receptor é variável no tempo após a emissão para compensar os efeitos da atenuação da propagação.

O problema da determinação da abertura de um feixe de ultra-som é complexo de quantificar dada a incerteza de inúmeras componentes, mas aqui fica de seguida uma ilustração do funcionamento de um sensor muito usado, o sensor ultra-sónico da Polaroïd (fig. 11). Um circuito electrónico força o transdutor a uma emissão de uma portadora de ultra-som de cerca de 50 kHz modulada por 16 impulsos quadrados. Após o último impulso, o circuito comuta de função e entra na fase de espera do eco; note-se mais uma vez que o transdutor é o mesmo. A partir desse momento, entra em acção um amplificador de ganho exponencial com o tempo, para compensar a atenuação do feixe ao atravessar o ar no trajecto de ida e volta. O

circuito vai acumulando a energia que retorna dos 16 pulsos emitidos até ao momento em que regressou energia suficiente para se declarar uma detecção. O sistema devolve então um valor numérico proporcional ao tempo decorrido entre a emissão e a detecção que pode ser assim usado para calcular a distância do sensor ao obstáculo reflector.

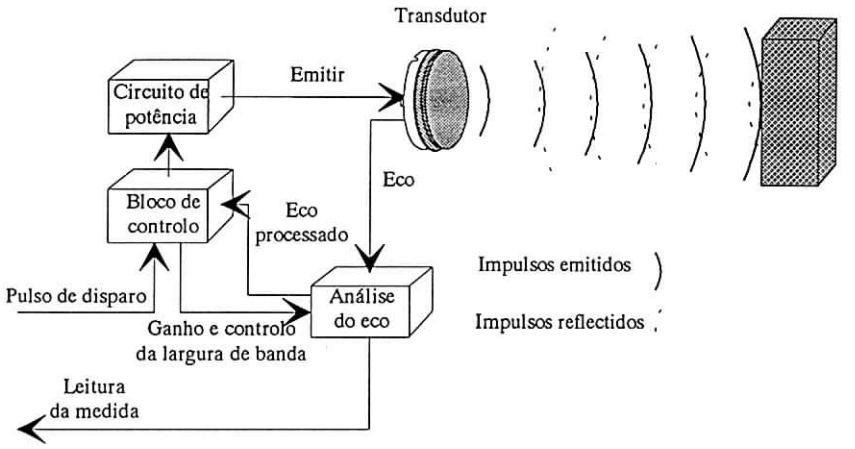


Fig. 11 - Diagrama de blocos do circuito da Polaroïd para o sensor de ultra-som

À parte o problema das reflexões especulares, que por si é já de elevada importância, há um outro problema associado ao uso simultâneo de múltiplos sensores: é o efeito da interferência mútua (*cross talk*).

Dada a escassa velocidade de propagação do som no ar (Eq. 8), o acto de medição é um processo "longo" no tempo.

$$C = 331.4\sqrt{\frac{T}{273}}\ m/s \qquad \textit{onde } T \textit{ é a temperatura em } \mathrm{K} \tag{8}$$

Nas situações de movimento de um robot, o recurso ao ultra-som enfrenta um grande desafio: efectuar medidas de distância em breve tempo para não haver dissociação entre as medidas e a real posição do robot. Se por exemplo uma medida necessita de 20 ms para ser efectuada (obstáculo a cerca de 3.5 metros), e se se esperar que uma medida termine para efectuar uma outra (quando se usam múltiplos sensores, obviamente) e se se possuírem 24 sensores, constata-se que é necessário um período de cerca de 480 ms para obter uma série completa de leituras. Se se pretender um erro de medida inferior a 5 % para essas distâncias (erro < 18 cm), o veículo não se poderá deslocar a mais de 37.5 cm/s. Se contudo se deseja um erro absoluto permanente sempre inferior a 10 cm (valor com interesse na prática), a velocidade máxima permitida ao robot desce para cerca de 20 cm/s. De qualquer forma, nenhuma taxa de disparo é suficientemente grande para garantir um erro mínimo: nos casos de rotação, e não só, a variação do ambiente vista por um sensor pode ser tão grande que a medida do sensor quando termina o ciclo de disparos pode ser perfeitamente irreal.

O disparo simultâneo de sensores surge como a solução para melhorar estes tempos de medição. E é então aqui que aparece o problema da interferência. Se os sensores trabalharem

na mesma frequência sonora (como é frequentemente o caso), haverá a necessidade de definir associações de sensores que possam ser disparados simultaneamente com mínima interferência, bem como estratégias para a sequência de disparo dos grupos de sensores.

Toda esta problemática prometedora não é contudo suficiente para dissuadir o interesse pelos sensores de ultra-som; chega-se ao ponto de os levar ao uso em funções bem complexas para a sua natureza como o caso referido atrás de tentar extrair as arestas verticais num dado ambiente por Crowley [Crowley89], e mais recentemente por McKerrow [McKerrow93b], ou mais ainda: tentar localizar e **identificar** determinados obstáculos dado o seu modelo à partida, e efectuando múltiplas medições com o(s) sensor(es) [Santamaria94].

1.5. Síntese e formulação da problemática da navegação

Após o exposto nos capítulos precedentes a propósito da navegação, são notórias algumas das questões que persistem nos sistemas de robótica móvel tradicionais, e que carecem de resolução ou pelo menos de progressos, e para os quais este trabalho se propõe contribuir. Fornece-se de seguida uma síntese dessa problemática numa lista com 5 pontos principais, aos quais se acrescenta um sexto ponto que, embora não tenha sido explicitamente abordado atrás e não seja visto como problema mas sim uma como preocupação actual, se relaciona com uma área que se reflectirá das propostas a apresentar para os problemas enumerados — as **arquitecturas de navegação**:

1) Necessidade de conhecer ou estimar a posição corrente para as acções de navegação.
2) Uma certa escassez de técnicas para operações de navegação autónoma baseadas exclusivamente nos sensores a bordo.
3) A maioria das acções de navegação são tomadas em função do destino final.
4) Poucos desenvolvimentos na representação do espaço livre nas imediações de um robot.
5) Dificuldades com os problemas associados a ultra-sons.
6) Importância (necessidade) de sistemas com propriedades reactivas na organização da informação sensorial.

A importância da maioria destes pontos dependerá de aplicação para aplicação, mas todos eles confluem de imediato num objectivo: o desenvolvimento de métodos para a navegação autónoma usando sensores de ultra-som.

O ponto 1 é uma realidade em quase todos os sistemas de navegação. Surge inclusivamente como a premissa base da navegação em si. Contudo, o conhecimento da posição pode não ser sempre possível, e durante esses momentos, o sistema deverá tentar subsistir autonomamente, daí a importância em saber prescindir do conhecimento da posição corrente, pelo menos temporariamente.

A navegação autónoma pode não ser uma exigência prática para todas as aplicações, mas todos os robots deveriam possuir um mínimo de capacidade de decisão local, e fazer recurso dos meios exclusivos a bordo nessas ocasiões. Motivos de segurança e independência do sistema justificam este ponto.

Uma larguíssima maioria dos sistemas de navegação existentes (excluindo os dispositivos guiados), executa os movimentos tendo em conta a posição mais ou menos exacta do local de destino. Verificar que se chegou ao destino não deveria ser feito em termos de coordenadas, pois pode nem sempre ser possível sabê-las ou verificá-las com exactidão. Por esta razão, o robot deverá estar habilitado a continuar as acções de navegação mesmo quando não haja destino definido. Este facto garante sempre uma última possibilidade de acção quando tudo o resto sucumbir (localização, comunicações, etc). Além desta vantagem como solução de recurso, uma tarefa de simples deambulação pode também ser o objectivo da navegação: basta pensar, por exemplo, num robot com funções de patrulha em circuito fechado num complexo de salas e corredores contíguos, como um museu, ou zonas de armazenamento de materiais valiosos!

Curiosamente, as técnicas tradicionais preocuparam-se com a representação do espaço numa sala, mas poucas vezes se tentou definir e representar o espaço instantâneo em torno de um robot móvel, sob o ponto de vista do próprio robot. Esta representação do espaço é crucial para as operações de navegação local (conceito que se explicará e estenderá), e permite ao veículo ter uma descrição útil do que é espaço livre ou espaço ocupado. Existem contudo trabalhos abordando este problema, mas não têm trazido soluções definitivas, suficientes e até práticas na representação do espaço livre em torno do robot, mesmo quando recorrendo a formas mais compactas e hierárquicas de estrutura de dados como, por exemplo, em [Noborio90].

Os problemas associados aos ultra-sons são muito conhecidos pela comunidade de robótica, e todos aceitam que as questões de interferência entre sensores e as reflexões especulares no ambiente comprometem seriamente a interpretação das medidas efectuadas. Também a solução para estes problemas carece de progressos.

Finalmente, mas não de importância menor, é a questão da reactividade dos sistemas robóticos às medidas sensoriais; esta questão não foi mencionada nos capítulos precedentes mas a sua importância é significativa quando se pretende utilizar a informação sensorial para acções concretas de navegação. É uma tendência, e simultaneamente uma tomada de consciência da comunidade robótica que o poder dos dados sensoriais deve ser maior e de acção mais directa e imediata sobre o comportamento do sistema. Deverão ser os valores sensoriais a gerar, ou no mínimo, "sugerir" as acções de movimento. Um exemplo extremo de reactividade (comportamento reflexo) é o seguinte: a iminência de uma colisão não deve fazer passar a decisão de paragem por um bloco de alto nível. Deve ser uma acção tomada

rapidamente e encarada como situação regular. Quanto menos excepcionalidade se impuser na interpretação das situações, mais simples e versátil resultará o sistema final.

Parte 2

Elementos para uma Proposta de Navegação Autónoma.

2.1. Introdução

Nesta parte propõe-se uma solução para as questões apresentadas na parte anterior. Serão definidos (ou redefinidos) conceitos a começar pela **Navegação Local**, e de seguida outros conceitos auxiliares tais como **Mapas de Percepção** e **Estratégias de Navegação (Local)**.

As propriedades dos Mapas de Percepção e as razões que determinam a sua topologia particular são também aqui tratadas, nomeadamente a estreita relação com a natureza dos sensores usados. Será também explicado o método para o cálculo desses mapas com base em redes neuronais e igualmente ilustrados outros dois métodos alternativos com considerações sobre vantagens e desvantagens mútuas. Relembre-se que os sensores usados para a percepção são unicamente sensores de ultra-som, e é com base exclusiva neles que se constroem os mapas de percepção.

Será também explicado como foram definidas as estratégias de navegação e delineados os métodos algorítmicos (ou não algorítmicos) que as implementam ou poderiam implementar.

Se bem que os resultados finais não sejam já expostos nesta parte, muitos dos resultados intermédios têm de ser mencionados para justificar segundas alternativas dentro de uma mesma técnica ou método.

2.2. Conceito de Navegação Local

Tradicionalmente, a navegação em robótica móvel define dois conceitos que resumem o próprio problema da navegação: planeamento de trajectórias (*path planning*) e desvio de obstáculos (*obstacle avoidance*). O primeiro está associado àquilo a que se pode chamar Navegação Global, isto é, calcular pontos de partida e chegada bem como possivelmente as trajectórias para o fazer, recorrendo ao conhecimento do ambiente.

O desvio de obstáculos é (tem sido, de uma forma geral) todo o conjunto das restantes fases necessárias para completar a tarefa de navegação. Contudo, este desvio de obstáculos não se consegue separar da execução da trajectória propriamente dita, isto é, usa elementos da Navegação Global para as suas próprias decisões (por exemplo, o ponto de destino final no caso dos **Campos de Potenciais** - *Potential Fields*). Na verdade, nos sistemas existentes mais divulgados, nem sempre é muita clara a distinção entre a navegação global e local.

Surge assim pertinente a necessidade de separar e distinguir estes dois níveis de navegação (Global e Local). Dessa forma, estes dois conceitos poderão ser desenvolvidos e enriquecidos de propriedades para levar a uma navegação mais autónoma e segura, principalmente no que respeita ao desvio de obstáculos em sentido estrito (isto é, sem pretender seguir necessariamente o destino final). Esta componente do desvio de obstáculos, neste último sentido, será assegurada pela Navegação Local.

Fig. 12 - Componentes da Navegação Local

Um sistema com possibilidades de existir de forma independente de uma unidade de Navegação Global, isto é, sem planeamento de trajectórias ou localização, deverá ser mais do que mero desvio de obstáculos como no caso dos **Campos de Potenciais**: terá forçosamente associadas a capacidade de avaliar o espaço livre no ambiente circundante, bem como estratégias de movimento dentro desse mesmo espaço (fig. 12). Em suma, não será um método orientado pelo destino final e pelos os obstáculos (*goal and obstacle driven method*) mas sim uma atitude de utilização do tal espaço circundante tendo em conta uma linha de comportamento assumida ou superiormente indicada: é assim um método orientado pelo ambiente (*environment driven method*) ou ainda, uma espécie de navegação referenciada ao próprio ambiente. Chamaremos a este paradigma de navegação: **Navegação Local**.

2.3. O robot e a distribuição de sensores de ultra-som

O robot a utilizar é uma plataforma móvel produzida pela companhia Robosoft: o *Robuter* [Robosoft91b]. Este robot consiste basicamente numa plataforma suportada em duas rodas motrizes e duas rodas livres auxiliares (fig. 13a). Numa cintura em torno da estrutura da plataforma, estão situados 24 sensores de ultra-som que serão usados para efectuar a avaliação

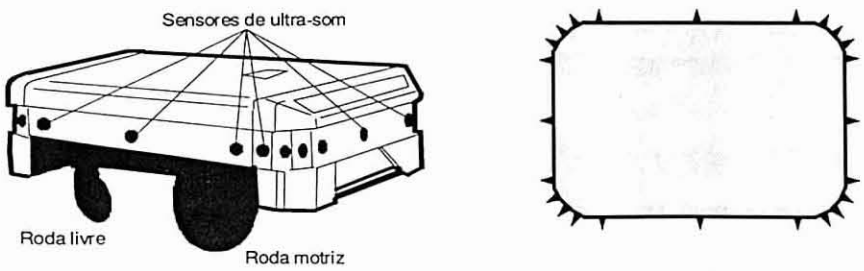


Fig. 13 - a) O *Robuter*. b) Distribuição da cintura de sensores no robot.

de distâncias (fig. 13b).

Como já foi descrito na parte I deste trabalho, o diagrama de radiação de um sensor do tipo usado (Polaroïd) não é simples, o que levanta problemas para a determinação de qual é a abertura α (fig. 14a) do feixe de ultra-som.

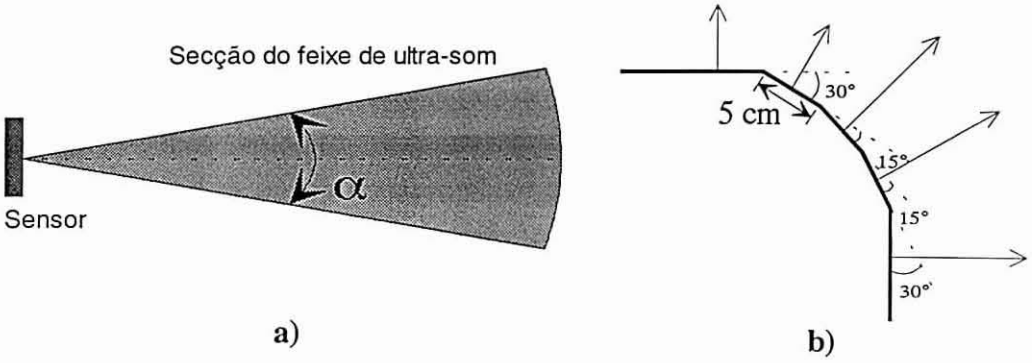


Fig. 14 - a) Secção planar do cone de emissão de um feixe de ultra-som. O ângulo α é a abertura do feixe cujo valor ronda os 20° [Ver texto]. b) Detalhe das orientações dos feixes nos cantos do robot.

Em primeiro lugar, o cone não tem uma abertura bem definida: a intensidade do feixe emitido varia com o azimute (ângulo medido relativamente ao eixo do feixe). Por outro lado, a quantidade de energia reflectida depende das propriedades de absorção dos obstáculos, e varia de forma não linear com a distância à qual ocorre a reflexão. Dadas estas dependências, não é fácil determinar teoricamente um valor prático para a abertura do feixe que possa ser usado

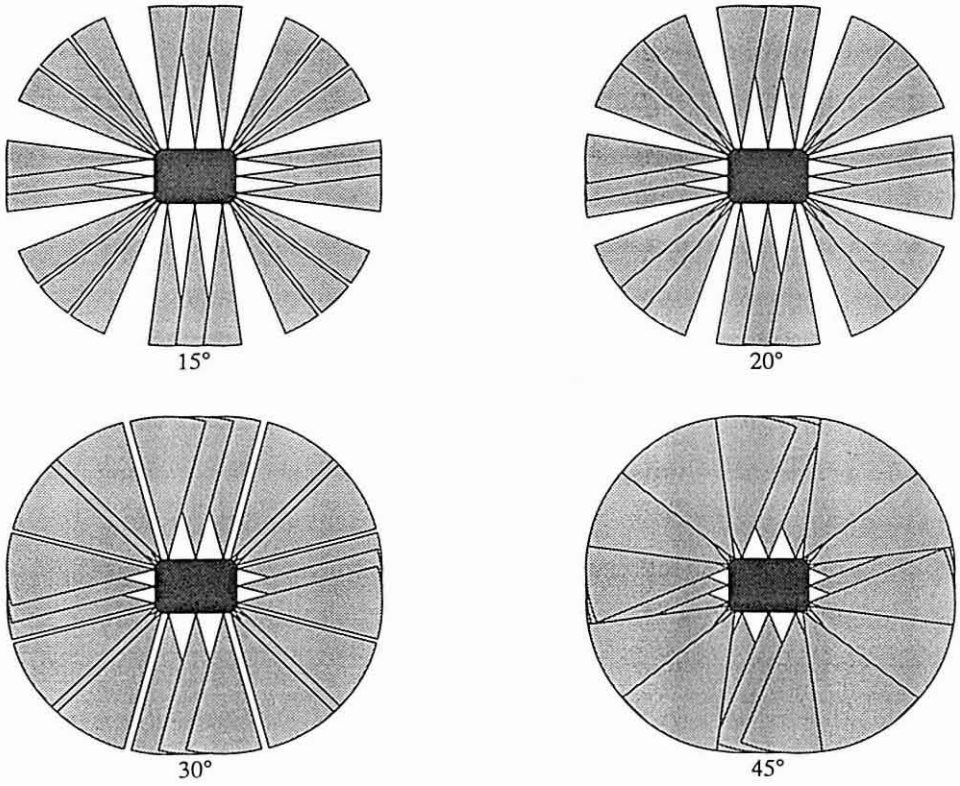


Fig. 15 - Exemplos teóricos do espaço coberto pelos sensores em função da abertura do feixe de ultra-som.

com plena confiança. Conjugando uma aproximação teórica e alguns testes experimentais [Santos93b], determinou-se que os ângulos reais se situam na gama dos 19°-20°; este valor é aliás também confirmado por estudos feitos pelo Instituto de Sistemas e Robótica e corresponde à amplitude do chamado **intervalo linear** [Moita94].

A distribuição dos sensores em torno do veículo implica uma determinada percepção do espaço, nomeadamente quanto à "quantidade" de espaço coberto pelos sensores. Esta percepção é assim função da separação entre sensores sucessivos bem como da abertura do feixe de ultra-som. A figura 15 ilustra essa dependência para um conjunto de diversas aberturas do feixe. Note-se ainda na figura 15, a existência de direcções "cegas" até aberturas de 30°. Isto equivale a dizer que se a abertura do feixe de ultra-som for menor que 30°, então existirá espaço que não será coberto por nenhum sensor do sistema. Todavia, estas direcções "cegas" não constituem um problema sério de segurança, uma vez que o robot nunca conseguirá efectuar movimento de translação ao longo destas direcções. Um potencial obstáculo, situado numa destas linhas, rapidamente sairá dela à medida que o robot se desloca (quer em rotação, quer em translação), caindo então no campo de visão de um ou mais sensores.

2.4. Avaliação do espaço livre - Mapas de Percepção

O primeiro passo para formalizar um método de navegação local é definir o processo de avaliação do espaço livre para as acções de movimento. Dever-se-á ser capaz de construir um mapa que traduza esse mesmo espaço livre de uma forma correcta, quer em termos de fiabilidade da representação, quer em maneabilidade na interpretação.

Obviamente, este mapa será obtido a partir dos dados sensoriais e não é excluída a possibilidade de integração temporal, isto é, de introdução de memória na construção de mapas sucessivos. A possibilidade de integração espacial (para o caso do robot em movimento) poderia ser considerada em algumas circunstâncias e, em princípio, dentro de um curto raio de acção, como se compreenderá pelas características do mapa; voltar-se-á a esta questão mais tarde.

2.4.1. Definição e conceitos

Um **mapa de percepção** é uma expressão que se usará para designar uma representação de uma certa grandeza ou propriedade em função das medidas de um ou mais dispositivos de percepção (os sensores). Uma representação destas implica uma dependência quer das sucessivas posições no espaço, quer, de certa forma, também do tempo, porque as medições podem apresentar variações mesmo em posição estática. Neste problema em particular, os mapas traduzirão a propriedade de ocupação de espaço em torno do robot através dos dados de distância dos sensores de ultra-som.

O mapa de percepção mais elementar que se pode construir com um conjunto de medidas de distância em torno de um robot, é definir um **polígono de espaço livre** em que os

vértices são dados pelos pontos do ambiente correspondentes às distâncias medidas pelos sensores (fig. 16).

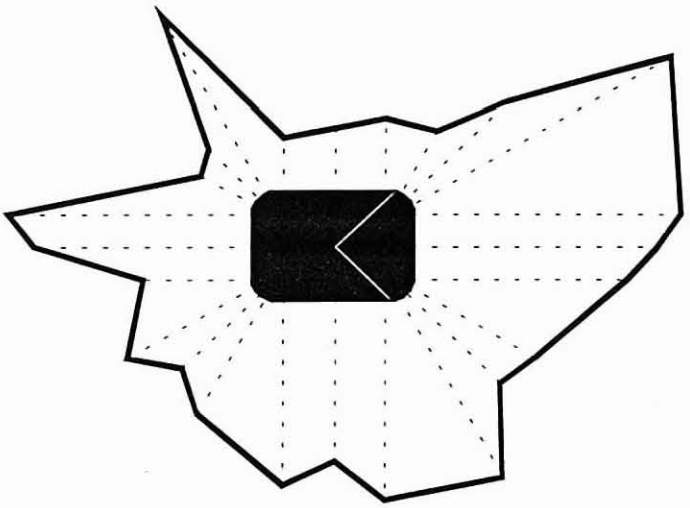


Fig. 16 - Um Mapa de Percepção elementar definido pelo polígono do espaço livre em torno do robot.

É de grande importância salientar que um mapa de percepção é uma estrutura de representação instantânea de uma propriedade espaço (ocupação) em torno do robot, e no referencial desse robot. Isto implica que um mapa de percepção não tem que ter relação alguma com o mapa global do ambiente. É um instantâneo (*snapshot*) tirado (necessariamente em linha de vista dada a categoria de sensores) de um determinado ponto do universo (posição do robot), em direcções à volta do robot. Este conceito de mapa de percepção é útil no aspecto em que permite representações do espaço independentemente do ambiente onde for usado.

O mapa de percepção definido pelo polígono de espaço livre (fig. 16), se bem que muito simples e imediato, apresenta problemas importantes. O primeiro é que, dado que cada sensor tem um papel independente na construção do mapa, variações em torno de uma mesma medida, ou falhas num dado sensor, implicam grandes variações na geometria do mapa. Estas variações gerarão polígonos bastante diferentes entre si, especialmente quando se pensar em termos da sua área (o espaço livre), ou numa qualquer espécie de direcção preferencial de espaço livre. Uma ilustração deste problema é, por exemplo, a oscilação entre um polígono convexo e um côncavo, resultantes de uma medida instável (no tempo) devido a situações limiares de especularidade.

A validade deste mapa reside também na inexistência de direcções "cegas", isto é, a percepção deverá garantir que as distâncias medidas pelos sensores representam a menor das distâncias aos obstáculos em torno do robot. Este problema está directamente relacionado com a resolução angular nas medições, ou o que é equivalente neste caso, a distribuição dos sensores e a abertura angular do feixe de percepção, uma vez que os sensores estão fixos ao robot. Para um reduzido número de sensores, ou para uma abertura do feixe demasiado pequena, estas direcções "cegas" podem ser importantes, mais uma vez porque a geometria do polígono de espaço livre poderá variar a uma taxa demasiado grande para um processamento

eficaz dos diferentes mapas. Ainda relacionado com a questão de resolução angular é o facto que a localização em direcção (localização angular) de um obstáculo usando ultra-som não pode ser feita com grande precisão. Deste modo, este tipo de mapas (polígono do espaço livre) é de difícil aplicação com dados de ultra-som, especialmente para um reduzido número de sensores.

Outro possível problema associado a este tipo de mapa advém da sua natureza contínua. Aqui, a questão será de ordem computacional: o mapa deve ser discretizado (numa grelha, certamente) para a análise e processamento sucessivos, exigindo recursos elevados, como se verá adiante. Dependendo um pouco da análise desejada, poderia não ser necessário reduzir o mapa a uma representação granular (grelha), donde o problema da manipulação computacional possa ser aparentemente minorado. De qualquer forma, o mencionado problema da total dependência da geometria do mapa com todos e cada um dos sensores, dificultaria a análise dos mapas para uma avaliação correcta do espaço livre em torno deste robot.

2.4.2. Grelhas e representação de mapas de percepção

A representação amostrada de determinadas grandezas (bidimensionais ou não) passa frequentemente pelo recurso a sistemas baseadas em um qualquer tipo de **grelha**, que, no fundo, se identifica com o sistema amostrador. A grelha é também um elemento de visualização, e a sua geometria, nomeadamente as dimensões das células, resultam dos critérios adoptados para a definição da resolução pretendida na discretização.

O tipo de grelha mais imediato, e frequentemente usado na resolução de alguns problemas em robótica (por exemplo [Elfes89] ou [Zelinsky88]), é a grelha cartesiana (fig. 17) Este tipo de grelha é muito usado também pelo facto das suas células serem inequivocamente identificáveis por simples indexação (uso do par ordenado com as coordenadas de um determinado ponto da célula — o centro, por exemplo).

Por outro lado, uma grelha cartesiana pode trazer desvantagens em determinadas condições. Um exemplo acontece quando se necessita de uma grande resolução da grandeza

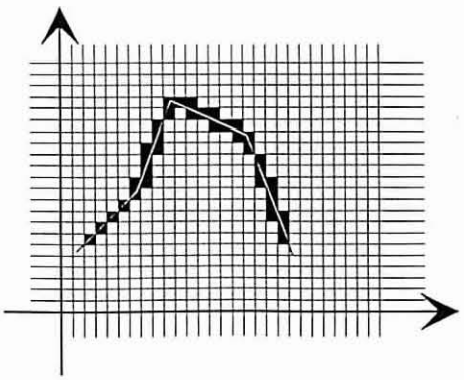


Fig. 17 - Uma grelha cartesiana: células rectangulares e iguais entre si.

discretizada: a consequência é o elevado número de células necessárias, e os problemas que isso pode representar ao nível da manipulação computacional (muita memória e maiores tempos de processamento). Se, por exemplo, se quisesse uma resolução de 10 cm num espaço planar rectangular de 7 m × 3 m (valores de utilidade futura), seria necessário usar uma grelha com 2100 células. Contudo, esta resolução poderia ter de melhorar para se ganhar definição nas linhas não paralelas aos eixos coordenados (fig. 18).

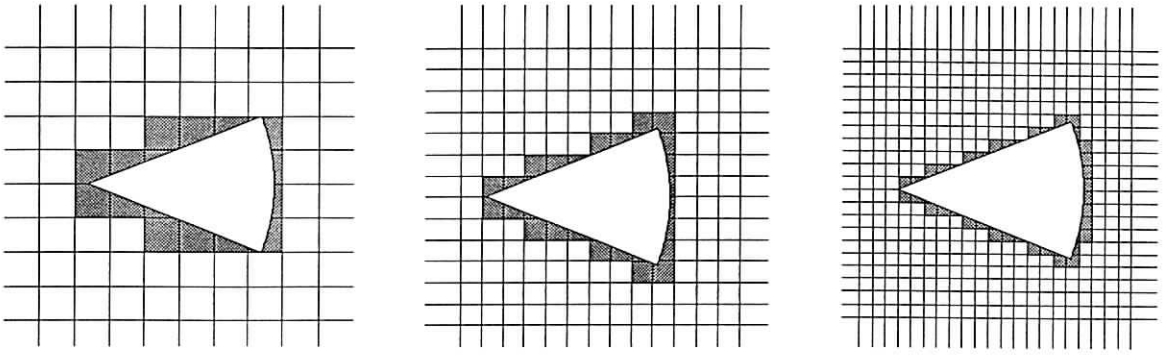


Fig. 18 - Diferentes resoluções de grelhas cartesianas.

Esta questão da existência de linhas não paralelas aos eixos, é de particular importância no problema de discretizar o espaço coberto por um feixe de ultra-som, ou seja, construir um mapa de percepção baseado numa grelha. As células que se situam na fronteira das regiões correspondentes aos cones, não podem ser simplesmente caracterizadas como pertencentes a uma ou outra região. Mais ainda, essas células poderão ter toda uma gama de valores de nível de inclusão numa das regiões, situações que, em rigor, devem ser previstas e manipuladas.

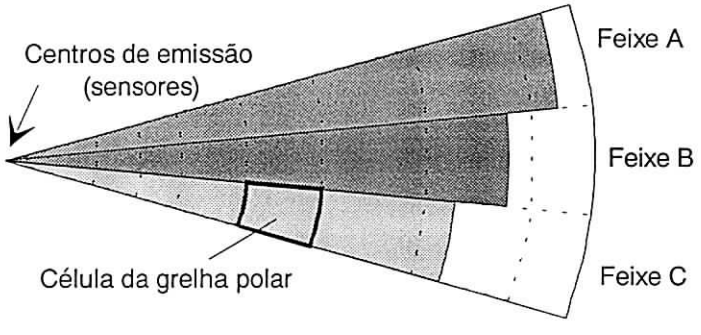


Fig. 19 - A grelha radial é adequada a representações polares da percepção com ultra-som. Este exemplo é de 3 sensores situados no mesmo ponto, mas devidamente orientados.

Uma solução óbvia para o problema das células nas fronteiras das regiões, é definir uma grelha cujas células obedeçam a geometrias conforme a das regiões a discretizar: sectores circulares e segmentos de coroas circulares (fig. 19). Esta representação radial (polar) é propícia para a representação de uma percepção feita a partir de um ponto de vista central: trata-se de **distâncias** medidas numa dada **direcção**, mas cobrindo um certo campo de visão correspondente à abertura angular associada ao processo de medição (como nos ultra-sons).

Como já foi mencionado, e nesta abordagem em particular, um mapa de percepção traduz o espaço livre e ocupado em função da direcção e distância dos elementos do ambiente que cerca o robot. Consequentemente, todo o espaço que separa um sensor de um obstáculo é definido como sendo **espaço livre**, e o espaço para além desse obstáculo é considerado **espaço ocupado**. A importância do espaço ocupado é, por si só, secundária, pelo simples facto que para além de espaço ocupado pode estar qualquer objecto, que é, à partida, indeterminado, logo inútil. Estar "para além de" significa neste contexto, estar na mesma direcção, mas a uma distância superior. As células de uma grelha que serve de suporte à construção de um mapa de percepção, terão associadas a si uma propriedade que é o estado de ocupação de espaço no local correspondente do ambiente.

A construção de uma mapa de percepção a partir de uma grelha em disposição radial (como na figura 19) apresenta de imediato as seguintes importantes propriedades:

- Maior facilidade em classificar as células em ocupadas e livres. De facto, desaparecem as ambiguidades de ocupação encontradas em grelhas cartesianas, isto é, não há intersecções "parciais" com as células da grelha, como na figura 18.
- O alinhamento radial garante uma **continuidade de profundidade** (*depth continuity*). Isto significa que para além de uma célula ocupada, não haverá células livres com significado: é de igual modo uma afirmação da coerência da ocupação do espaço numa dada direcção.
- O ajuste de resolução da grelha não aumenta o número de células nas fronteiras das regiões, facto que é importante em termos da atribuição do valor de ocupação às células.

Uma propriedade que não aparece enumerada pela simples razão que não é uma consequência casual, mas sim uma força impulsionadora desta ideia, é que esta geometria é perfeitamente conivente com as propriedades físicas dos sensores de ultra-som (isto é, com as propriedades do tipo de medição proporcionada).

Em termos práticos, a resolução angular nestas grelhas é fixa e, em geral, ditada pelos princípios físicos inerentes ao método (a abertura de um feixe de ultra-som é a máxima resolução angular que se pode ter). A resolução linear pode ter a ver com mais do que apenas factores da precisão desejada das distâncias, como se verá adiante.

Na prática, a localização dos sensores não é como indicado na figura 19; no robot, os sensores estão afastados entre si, e as direcções dos seus eixos de emissão não se ajustam necessariamente à abertura dos feixes. Assim, essa grelha perfeitamente polar é insuficiente para representar a informação gerada por esta distribuição de sensores (fig. 13b). Uma observação mais atenta da distribuição dos sensores (e do conjunto de aberturas dos feixes, que doravante se supõe ser de 20°) poderá induzir a ideia que os sensores podem ser agrupados com base na sua separação e orientação, consequentemente com base na zona de

espaço comum que cobrem (fig. 20). Estamos em presença de um conceito (e também propriedade) importante em desenvolvimentos que se seguem mais à frente: a **redundância sensorial**.

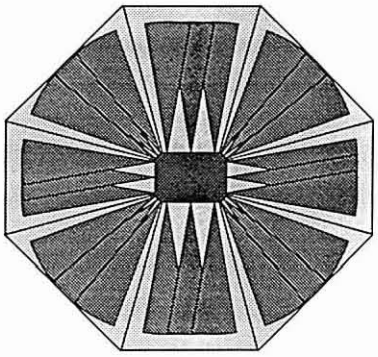


Fig. 20 - Associação dos 24 sensores em 8 grupos tendo em conta a zona de espaço comum coberta.

Estes agrupamentos definem assim 8 regiões trapezoidais em torno do robot. Cada uma destas regiões poderá constituir um submapa de um mapa geral de percepção. Uma região é coberta por 3 sensores, mas não é de excluir à partida, que sensores de outras regiões influenciem o seu submapa de percepção; antes pelo contrário, os feixes de ultra-som, em determinadas condições, podem produzir detecções bem além dos 20° determinados para situações médias.

Uma grelha para servir de suporte a mapas para representar regiões deste tipo (regiões trapezoidais cobertas por múltiplos sensores como na figura 20) tem de ser mais complexa que as simples grelhas polares definidas atrás. A coerência entre as medidas sensoriais e a sua representação no mapa deve ser sempre mantida em observância. Como consequência, foi necessário definir um conceito mais geral de grelha que inclua os tipos anteriores: as **Grelhas de Geometria Geral**, para as quais se adopta a sigla **GGG**.

2.4.2.1. Grelhas de Geometria Geral (GGG)

A definição das GGG é feita de forma relativamente simples, sem recorrer necessariamente a uma axiomática complexa, mas assenta num conjunto de princípios que se enumeram de seguida:

- Uma GGG é uma estrutura hierárquica (finita) de células, com células progenitoras e células descendentes.
- Os limites geométricos da grelha definem a primeira célula, e é a única que não possui progenitor — é o universo do problema (a região ou mapa).
- Uma célula é um hiperpolígono generalizado, isto é, os elementos que constituem a sua fronteira são qualquer ente de dimensão geométrica imediatamente inferior à da célula (as células planares [polígonos generalizados a 2-D] são limitadas por linhas, e as células volúmicas [3-D] são limitadas por superfícies [planos, etc..])

- Cada célula tem um determinado número de descendentes associado às regras de fragmentação. A célula pode não ter descendente algum, nesse caso é uma célula terminal e constitui um ramo final na árvore hierárquica. As regras de fragmentação podem incluir, entre outros, o número de descendentes e os seus tipos. Este princípio é consequência do primeiro princípio que postula uma hierarquização da estrutura.
- Cada célula é de um determinado tipo, e caracteriza-se pelas regras de fragmentação.

Estes princípios, que foram maioritariamente impostos por questões de ordem prática, permitem a definição de uma grande variedade de grelhas finitas: permite todo o tipo de anisotropias no que respeita ao número de células, suas dimensões e geometria, e também à sua densidade, ou seja, permite até possivelmente uma distribuição de células onde o seu número e dimensões dependa da direcção.

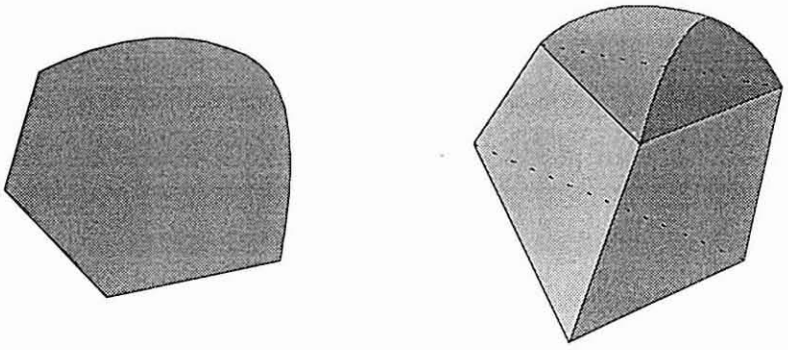


Fig. 21 - Exemplos de polígono e de poliedro generalizado. Note-se a presença de linhas e superfícies curvas respectivamente a par dos segmentos de recta e planos.

Uma simples grelha cartesiana monométrica no plano é uma GGG em estrutura de *quadtree*, em que a primeira célula é um quadrado. É evidente que na prática poucas seriam as vantagem em gerir um grelha cartesiana desta forma; isso é vantajoso quando as grelhas que se pretendem usar são anisotrópicas por motivos inerentes ao problema, e onde a referência das células não é feita pela simples indexação tradicional. Para este problema em particular as células serão bidimensionais e conformes às necessidades de representação a duas dimensões. Todavia, o conceito é generalizável para, por exemplo, construir um mapa de percepção a três dimensões associado à extremidade de um braço robótico e solidário com ele!

A definição apresentada para as GGG não contempla a possibilidade de uma grelha de dimensões variáveis, isto é, que aumente de dimensões globais (a primeira célula deveria ser definida doutro modo, e a par de regras de fragmentação deveriam ser definidas também regras de agregação).

As GGG que se usam neste problema em particular são a duas dimensões, e doravante todas as referências a GGG serão desse tipo, salvo indicação no momento.

2.4.2.2. Relação entre os dados sensoriais e a GGG a definir

Foram já assinaladas duas propriedades das grelhas polares que resultam vantajosas no tipo de mapa de percepção pretendido. Essas propriedades mantêm-se nesta generalização de grelhas que são as GGG. As premissas e condições suplementares para a concepção de uma GGG adequada ao problema da construção de um mapa de percepção usando dados sensoriais no sistema (robot e sensores) descrito anteriormente, e ilustrado sucessivamente nas figuras 13, 15 e 20, são:

- Minimizar o número total de células da grelha por motivos computacionais.
- Definir as dimensões úteis da grelha para as acções de navegação local.
- Definir as células de modo que o espaço a que correspondem seja coberto pelo maior número possível de sensores. Isto pretenderá compensar as falsas medições de sensores isolados.
- Ajustar a densidade de células e, consequentemente, as suas dimensões, de forma conivente com a fiabilidade dos dados sensoriais.

A minimização do número total de células na grelha é consequência da tal redução das exigências computacionais (que foi também uma das razões pela não preferência das grelhas cartesianas). Uma outra razão de ordem computacional está em aproximar os arcos de circunferência das células de uma grelha de tipo radial pela respectiva corda como se ilustra na figura 22.

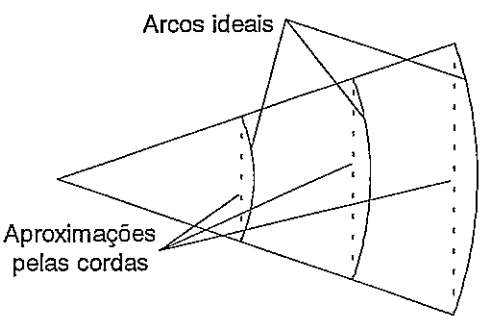


Fig. 22 - Aproximação de arcos de circunferência por cordas em células de grelhas de tipo radial.

Com um mapa de percepção, pretende-se traduzir leituras sensoriais em bruto numa representação que possa utilizar vantajosamente a redundância sensorial, mas sem implicar uma redução da informação original. Um elemento de alguma importância que convém não descurar é a resolução angular. Se cada uma das oito regiões da figura 20 não for fragmentada longitudinalmente (para maior resolução angular), haverá uma perda de resolução angular, dado que três sensores foram reduzidos a uma situação como se fossem um só!

A definição dos limites da grelha é quase arbitrária, mas há um conjunto de factores práticos que deve ser tomado em conta. Esse factores incluem i) a velocidade máxima que se espera usar, ii) o espaço lateral necessário é menor do que nas direcções de possível movimento (frente e retaguarda), iii) quantificar o espaço livre a grandes distâncias (mais de

3-4 metros) envolve grandes riscos devido a reflexões especulares e iv) a navegação local, como é a sua pretensão na prática, lida com espaços livres na ordem de grandeza das dimensões do próprio robot (1-2 vezes), ou seja, o que pretende é procurar novo espaço para onde poder levar o veículo.

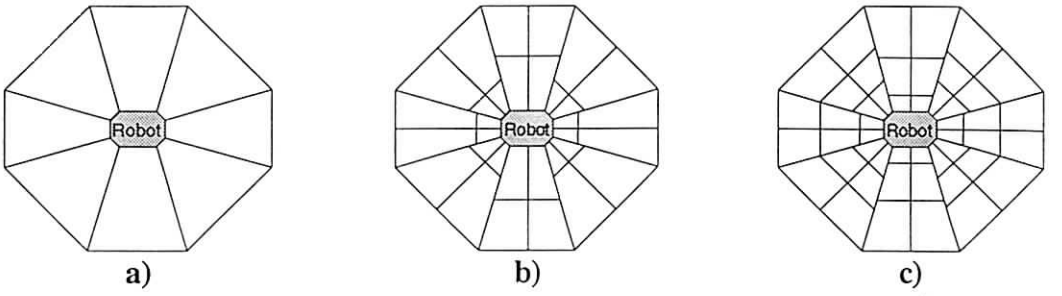


Fig. 23 - Ilustração de uma sequência de passos na fragmentação de uma GGG para mapas de percepção com o *Robuter*. a) Os 8 trapezóides iniciais. b) Fragmentação dos trapezóides em 4 partes. c) Fragmentação de alguns dos subtrapezóides.

A figura 23 exemplifica a elaboração de uma grelha respeitando algumas das directivas estipuladas. Note-se nessa figura um facto importante: relembrando a definição de Grelha de Geometria Geral, a primeira célula foi fragmentada em 9 partes, 8 trapezóides que incluem todo o espaço à volta do robot, e, no centro, um octógono irregular que corresponde ao espaço do próprio robot! Esta célula não tem grande interesse prático: apenas completa a definição formal da grelha, e é útil para os algoritmos de manipulação geométrica.

Uma regra empírica geralmente invocada por quem trabalha com sensores de ultra-som em robótica, é que as medidas curtas são mais fiáveis do que as longas. Não porque exista uma variância na precisão do sistema de medição, mas porque as reflexões especulares (que correspondem a medidas falsas) originam medidas maiores do que a distância real ao obstáculo. Esta regra sugere que uma quantificação do espaço livre se faça a diferentes resoluções consoante a distância desses pontos do espaço aos sensores: eis desta forma uma primeira directiva para o cumprimento das premissas expostas atrás. Esta directiva implica que as células menores se encontrem junto ao robot, e as maiores nos limites exteriores da grelha.

As dimensões exactas das células podem obedecer a vários critérios, mas é razoável, para qualquer um deles, definir as células mais próximas do robot com comprimento comparável à menor distância passível de ser medida com os sensores de ultra-som. Comprimentos menores dessas células são inúteis, e comprimentos maiores equivalem a perda de resolução na zona mais importante do espaço em torno do robot. Mencionam-se de seguida dois possíveis critérios para a repartição das células restantes em cada uma das oito regiões indicadas:

1) Usar os pontos de intersecção dos feixes duma mesma sub-região (submapa) como delimitadores de células.

2) Seguir uma dada progressão para o cálculo do comprimento das células numa sub-região (este foi o método escolhido).

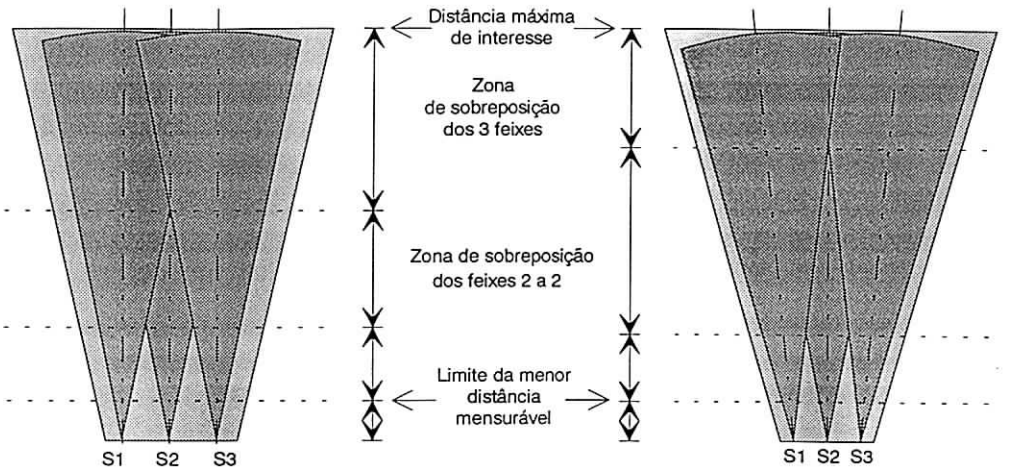
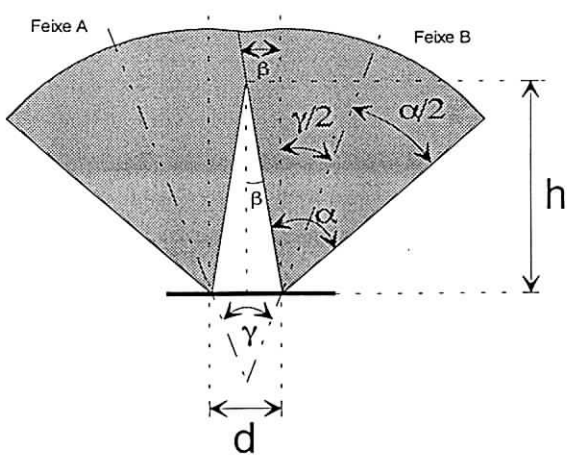


Fig. 25 - Zonas de intersecção dos 3 feixes de ultra-som dentro de subgrupos da GGG: à esquerda uma situação onde os feixes são paralelos, e à direita onde não o são (cantos do robot). NB: trata-se de uma ilustração exagerada da situação real!

A possibilidade do método 1 compreende-se pela análise da figura 25 onde são ilustrados os pontos de intersecção dos feixes dentro de uma mesma sub-região da Grelha de Geometria Geral. Isto traduz as zonas de sobreposição das influências dos sensores, invocando de novo a propriedade da **redundância sensorial** no espaço. Como se pode ver na figura 24, os limites das zonas de sobreposição (**h**) dependem geometricamente de 3 factores: a abertura dos feixes α, a separação entre sensores **d**, e a orientação relativa entre os feixes γ.

Um cálculo rápido, assumindo aberturas de feixe em 20°, dá como limites das células dentro cada grupo de sensores o exposto na tabela 1.



$$\frac{\alpha}{2} = \beta + \frac{\gamma}{2}$$

$$\beta = \frac{\alpha - \gamma}{2}$$

$$\frac{h}{\frac{d}{2}} = \cot g(\beta)$$

$$h = \frac{d}{2}\cot g\left(\frac{\alpha - \gamma}{2}\right)$$

Fig. 24 - Cálculo dos pontos de intersecção de dois feixes de ultra-som

	Grupos de frente e trás	Grupos dos cantos	Grupos laterais
Separação entre sensores adjacentes/não adjacentes (cm)	21.5 / 43.0	5.0 / 10.0	34.0 / 68.0
Orientação de feixes entre sensores adjacentes/não adjacentes	0° / 0°	15° / 30°	0° / 0°
Distância do robot ao ponto de intersecção de cones de sensores adjacentes (cm)	**82**	**57**	**129**
Distância do robot ao ponto de intersecção de cones de sensores não adjacentes (cm)	**163**	Sem significado para abertura de 20°	**258**

Tabela 1 - Concretização de valores para as limites longitudinais das células em função da separação e orientação dos sensores em cada grupo de 3 sensores.

Note-se que os valores calculados (na tabela 1) para os pontos de intersecção (sobreposição) dos feixes de ultra-som são meramente indicativos. O seu significado teórico é importante porque representam os limites da máxima fiabilidade que se poderia extrair de cada um dos 8 grupos de 3 sensores. Porém, a sua aplicação na prática (nomeadamente na navegação, como se verá adiante) comprometeria seriamente a simplicidade e lógica dos métodos que os utilizariam. A primeira e maior razão é que a diferença de dimensão longitudinal de células correspondentes em grupos diferentes, geraria incoerência de avaliação de espaço livre, como se pode perceber na figura 26.

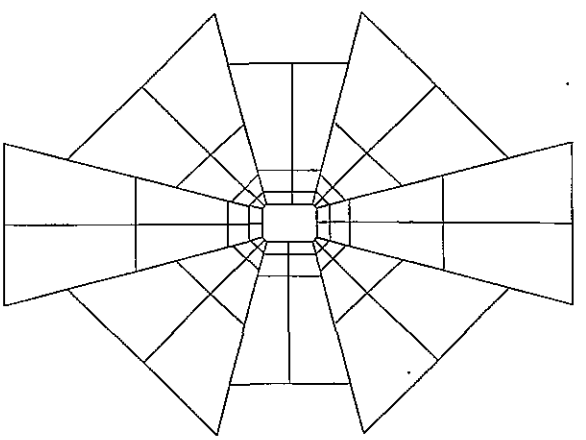


Fig. 26 - Possível GGG cujas células obedecem a um primeiro critério de divisão.

A primeira célula de cada grupo (a que está junto ao robot), como já foi mencionado, terá uma dimensão relacionada com a distância mínima que se pode distinguir com um sensor de ultra-som nestas circunstâncias. Essa distância não é fixa, e depende de sensor para sensor; verificou-se experimentalmente que ronda aproximadamente os 12 cm, consoante alguns ajustes individuais de cada sensor. Desta forma, foi estipulado uma dimensão longitudinal de 12 cm para as primeiras células de cada grupo.

Uma vez que o método 1) para a definição das dimensões das células apresenta desvantagens na aplicação prática em vista, optou-se pelo método 2), mas respeitando sempre as premissas impostas. Tentou-se então chegar a uma solução que viabilizasse o uso da grelha assim definida para as acções de navegação que adiante se descreverão.

Tendo em conta todos os elementos apresentados, a grelha final, que se ilustra na figura 27, é caracterizada pelos seguintes pontos:

- A grelha consiste em 8 subgrelhas cada qual cobrindo uma determinada região do espaço em torno do veículo. Esta subgrelhas têm o formato trapezoidal e são subdivididas longitudinalmente; são ainda subdivididas transversalmente para definir células às seguintes distâncias: 0-12 cm, 12-45 cm, 45-130 cm e 130-350 cm.
- As 8 subgrelhas pertencem a um de 3 tipos apenas. Esta escassa variedade de tipos é explicada pela simetria do robot (os cantos, os topos e os lados).
- A grelha nas direcções laterais é mais pequena do que nas restantes direcções. Isto deve-se à desnecessidade de ter uma representação extensa em direcções onde, por um lado, não há possibilidade de movimento de translação, e por outro lado, a densidade sensorial decai e pode comprometer a interpretação dos dados dos sensores.

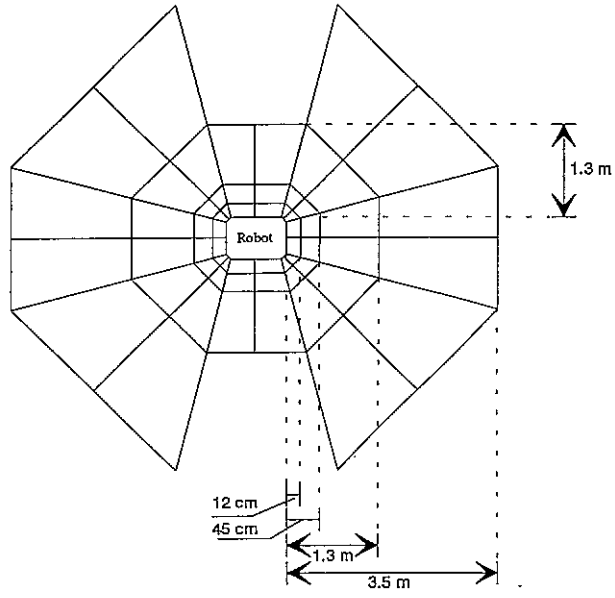


Fig. 27 - Dimensões e distribuição das células na grelha final.

A regra usada para o cálculo dos valores das dimensões não é muito precisa: pretendeu apenas que, dentro de uma dada região, cada célula tivesse um comprimento maior que a célula anterior, começando a contar na célula junto ao robot. Não existe uma razão linear de crescimento: esse seguiu uma regra que resultou em razões entre 3 e 4 por motivos ligados à implementação prática das estruturas de dados associados a esta grelha (nomeadamente as regras de fraccionamento sucessivo das células). Na realidade, o resultado final aproximou-se,

de forma não intencional, de um crescimento exponencial (quiçá mais adequado) para a dimensão longitudinal das células, como se pode depreender na figura 28.

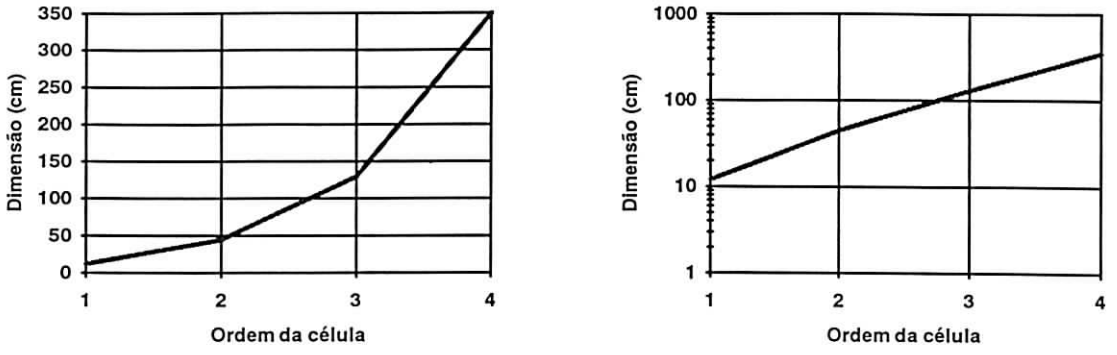


Fig. 28 - Dimensão longitudinal de uma célula de acordo com a sua ordem no trapezóide: a célula 1 é a mais próxima do robot. À direita uma representação com ordenadas logaritmizadas para ilustrar como o crescimento das células na grelha final se aproximou de uma lei exponencial.

2.5. Elaboração de mapas usando os dados sensoriais

Elaborar um mapa de percepção instantâneo significa, neste caso, transformar um conjunto de leituras sensoriais numa representação o mais correcta possível do espaço livre em torno do robot. Pretende esse facto suplantar, ou pelo menos minimizar, os erros existentes nesses dados sensoriais, que no caso de sensores de ultra-som ocorrem com grande frequência e sem possibilidades de corrigir individualmente cada sensor. A técnica dos mapas de percepção irá, por conseguinte, tirar partido da redundância espacial dos sensores e chegar assim a essa desejada representação do espaço.

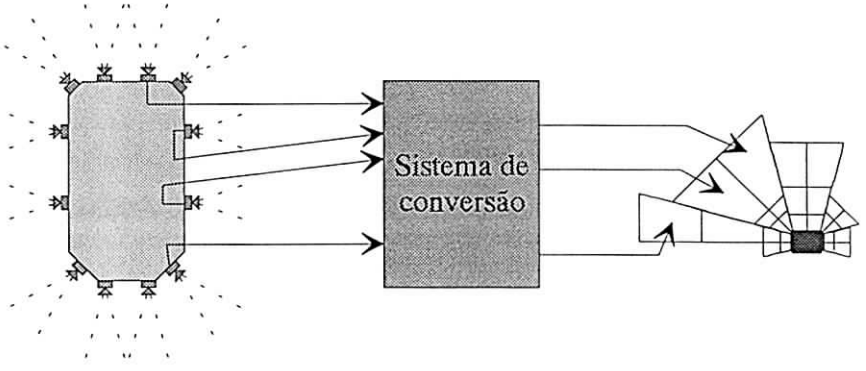


Fig. 29 - Ilustração do princípio da conversão de leituras sensoriais num mapa de percepção.

Calcular um mapa a partir dos dados sensoriais surge agora como o problema fundamental: consiste, grosseiramente, em "fundir" um conjunto de medidas sensoriais e preencher de forma coerente todas as células de uma GGG resultando assim no mapa de percepção (fig. 29). No fundo o problema subdivide-se em 3 componentes principais:

- Transformação da representação dos dados — leituras sensoriais em mapas.
- Redução (simplificação) dos dados — dados mais fáceis de interpretar.
- Correcção de eventuais falhas em dados — sensores faltosos.

Sistemas que prometem poder desempenhar essas funções com sucesso (entre outras razões também pelos seus resultados em trabalhos no passado), são aqueles à base de redes neuronais (Ver o apêndice **Conceitos sobre redes neuronais**). Contudo não se deixará de testar outras alternativas para este sistema de conversão, como se verá mais tarde.

2.5.1. Uso de Redes Neuronais

A opção pela redes neuronais justifica-se por algumas propriedades que se adequam a um problema desta natureza, nomeadamente:

- Capacidade de estabelecer a relação entre conjuntos de dados, mesmo que complexa e/ou não linear, através de um processo de aprendizagem.
- Grande capacidade de generalização (inter e extrapolação) para dados desconhecidos, inesperados ou, em alguns casos, até fora da gama de treino. Esta acção traduz um certo efeito de filtragem e inclusive de integração (fusão) de dados.
- Uma rede neuronal tem, em geral, uma velocidade de operação independente dos dados (fixa), e muita rápida (consiste essencialmente na multiplicação de matrizes).

Destas propriedades, as duas primeiras são as mais significativas para o problema que é proposto resolver. Relembremos, a propósito, uma afirmação de Anderson e Rosenfeld [Anderson88], que caracteriza de modo informal uma característica fundamental das rede neuronais:

> "[...] as redes [neuronais] são boas a detectar regularidades estatísticas, e não se incomodam com inconsistências ocasionais [...]"

Note-se ainda que as redes neuronais têm a capacidade de integrar dados de formatos e origens diferentes; isto é, podem ser usadas para, por exemplo, operar sobre dados oriundos de sensores diferentes no seu princípio (dados de distância e intensidade de iluminação numa imagem vídeo, ...). Este facto, se bem que não tenha relevância imediata para este problema, poderá ser importante em desenvolvimentos futuros deste trabalho.

No que respeita ao sucesso das redes neuronais no passado, é fácil indicar vários exemplos disso mesmo. Fora da robótica, há muitos casos já quase "clássicos", não fosse a sua tão recente data: os primeiros exemplos significativos (não necessariamente os melhores até aos tempos actuais) para reconhecimento de escrita humana, por exemplo, deram-se com L. Jackel *et al.* [Jackel88] que usaram um circuito VLSI de 54 neurões para proceder ao processamento das "imagens" dos dígitos, e Y. LeCun *et al.* [LeCun90] que se dedicaram ao reconhecimento dos algarismos dos códigos postais, usando um conjunto de mais de 9000 dígitos extraídos de envelopes passados nos serviços postais dos Estados Unidos. No caso de LeCun foi usada uma rede com múltiplas camadas escondidas (*hidden layers*) e treinada com **retropropagação** (*back-propagation*).

Dos possíveis múltiplos exemplos ainda restantes, é de notar o trabalho de Gorman e Sejnowski, também por uma certa afinidade com a situação que se trata neste trabalho. Gorman pretendia classificar ecos de ultra-som reflectidos no fundo do mar como sendo ecos de rochas ou ecos de cilindros de metal [Gorman88]. Também neste trabalho o algoritmo de retropropagação foi usado, e as suas prestações revelaram-se melhores que outro tipos de classificadores (como o "**vizinho-mais-próximo**" (*nearest neighbour*) ou o "**classificador Bayesiano**").

2.5.1.1. Usos anteriores de redes neuronais em robótica

A ideia de aplicar redes neuronais em problemas de robótica não é inteiramente nova. Contudo, essas aplicações deram-se, ou têm-se dado, em questões específicas diferentes desta aplicação em particular.

Existe uma certa abundância de exemplos no campo da robótica "fixa", ou seja os braços articulados. As questões que aí se abordaram têm a ver com a inserção de redes neuronais nas malhas de controlo, para resolução da cinemática inversa, por exemplo. Muitos exemplos do uso de redes neuronais em robótica, com predominância dos braços articulados, podem ser encontrados na colectânea de artigos editada por Bekey e Goldberg [Bekey93].

No âmbito do projecto Esprit ANNIE (*Applications of Neural Networks for Industry in Europe*) [ANNIE91], encontram-se trabalhos (a nível de simulação) onde as redes neuronais são utilizadas para guiar um veículo num ambiente mais ou menos estruturado. Para o efeito, usam dados sensoriais de ultra-som. É de salientar o carácter restrito deste trabalho; dado tratar-se de simulação, alguns problemas reais não são abordados com grande importância: o exemplo mais notório é a fiabilidade absoluta em leituras individuais dos sensores de ultra-som, que como se mostrou, não é uma política segura para avaliar o espaço ao redor do veículo.

Talvez o exemplo de mais sucesso em robótica onde se usam redes neuronais seja o projecto ALVIN da Universidade de Carnegie Mellon (CMU). Com efeito, Dean Pomerleau desenvolveu um sistema de navegação que também usa redes neuronais para uma avaliação do espaço onde é possível a locomoção. Não se trata de uma análise apenas de quantidade de espaço livre em torno do robot, mas inclusive da interpretação de imagens captadas por uma "retina" CCD [Pomerleau93]. Deve-se todavia fazer notar uma certa especificidade do tipo de navegação proposta por Pomerleau: é sobretudo uma navegação "macroscópica" e essencialmente de exterior — o veiculo circula pelos seus meios, e com relativa facilidade, em caminhos, estradas e auto-estradas.

2.5.1.2. Redes neuronais específicas para este problema

As funções propostas para uma rede neuronal para este problema, são as de ensinar o sistema a fazer uma correspondência entre as medidas sensoriais e a real ocupação do espaço

em torno do robot. É de sublinhar que a função de um sistema neuronal não é definir um sistema que reage espontaneamente ao espaço livre (ou ocupado), mas a criação de uma representação para esse mesmo espaço. O comportamento, ou as reacções ao espaço livre, serão feitos posteriormente a um nível diferente, e por um outro sistema independente deste "construtor" de mapas.

A rede devidamente treinada manter-se-á, em princípio, invariável durante a fase de operação; não é, portanto, uma rede que aprende continuamente durante a fase de operação. O ensinamento da rede é, para já, feito com exemplos concretos: a fase de aprendizagem, que é feita *off line*, consiste em fazer a rede associar conjuntos de leituras sensoriais a valores de células ocupadas e livres, que, por exemplo, um operador humano define por observação factual (fornecendo desta forma a informação correcta a ser aprendida). Isto é diferente de uma rede puramente reactiva, do tipo de aprendizagem reforçada (*reinforcement learning*) como por exemplo nos trabalhos de J. Millán [Millan92], onde nenhum ensinamento concreto é feito, apenas se penalizam ou reforçam as acções conforme o seu êxito durante a própria fase de operação que se confunde com a fase de aprendizagem. Na robótica, a distinção ou não destas duas fases (aprendizagem e operação) é, por vezes, invocada na definição da **aprendizagem de robots** (*robot learning*) [].

2.5.1.3. Ensinamento da rede: os dados de entrada e os dados pretendidos

Os dados para o treino das redes devem ter uma representação apropriada: a situação mais usual é a de os normalizar em uma de duas possibilidades: no intervalo [-1,+1] ou [0,+1]. Está-se a pensar, claro, num universo contínuo de entradas possíveis; em alternativa, o recurso a padrões de entrada binários usa quase sempre os conjuntos {-1,+1} ou {0,1}. Estas duas possibilidades, em ambos os casos, são equivalentes: nalgumas implementações (simuladores de redes neuronais) usa-se a primeira, noutras, usa-se a segunda. Os motivos estão relacionados com questões de programação ou representação preferida de dados: possivelmente questões de gama dinâmica finita na implementação computacional podem justificar uma ou outra opção.

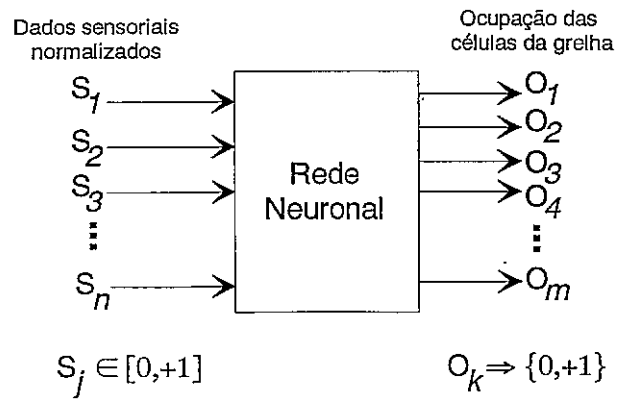


Fig. 30 - Formatação dos dados de, e para a rede neuronal neste problema.

Neste problema, o que se pretende que a rede aprenda é gerar todos os valores de ocupação das células de uma dada grelha. Se a uma célula ocupada se fizer corresponder o valor 1, e uma célula livre o valor 0, o problema traduz-se então em tentar estabelecer uma relação entre valores normalizados das medidas sensoriais e uma lista de valores binários de ocupação de um conjunto de células (grelha):

Na figura 30 os valores de saída da rede neuronal estão indicados como forçados a apenas dois valores através de um qualquer processo de limitação (*thresholding*). Para já, é irrelevante se esse processo ocorre dentro da própria rede neuronal (unidades de saída com função de activação adequada), ou se é feito numa fase de pós-processamento dos dados gerados pela rede.

2.5.1.4. Dados reais para o treino supervisionado das redes

As redes neuronais podem ser classificadas em duas grandes categorias quanto à sua aprendizagem: as de **aprendizagem supervisionada** e as de **aprendizagem não supervisionada.** Para o treino das primeiras é necessário dispôr de um conjunto de exemplos. Um exemplo é, neste contexto, um par ordenado com um padrão (vector) de entrada, e a respectiva correspondência (padrão, classe ou vector de saída) segundo a função ou lei que se pretende que a rede venha a aprender. Estes exemplos são angariados, em geral, através de procedimento experimental. São ilustrações desta categoria, as redes de Hopfield, os Perceptrões, as *Adalines*, e em geral, todas as redes treinadas com o algoritmo de retropropagação, entre muitas outras.

Em contrapartida, as redes de aprendizagem não supervisionada não carecem de exemplos para efectuar a sua aprendizagem. A maioria dos casos usados está relacionada com o uso de regras de aprendizagem competitiva, e são redes dedicadas à (re)organização ou classificação natural de informação. Quer isto dizer que uma rede deste tipo "aprende" a agrupar os dados de entrada (vectores) em classes. Cada saída da rede é, em geral, uma classe. À partida, o operador determina (impõe) o número de classes que espera na distribuição dos seus dados e depois fornece os dados à rede, segundo uma ordem perfeitamente aleatória. A rede "aprende" a separar os dados de acordo com a sua natureza matemática ou topológica seguindo regras de competição entre unidades de processamento da rede (neurões). Um exemplo destes algoritmos extraordinários são os mapas auto-organizativos de Kohonen (*Kohonen Self-organising Feature Maps*) [Kohonen89]. Estas redes são de treino bastante moroso (aliás, tal como as treinadas com retropropagação), e requerem uma grande quantidade de dados apresentados de forma uniformemente distribuída (representativos e ponderados relativamente à abundância nas classes). Uma curiosidade relativamente a estes mapas de Kohonen é que não há garantia que a função de classificação evolua correctamente ao longo do treino; se inclusivamente aos pesos aleatórios iniciais da rede não impuser uma certa

tendência (não sendo totalmente aleatórios) a função pode evoluir de formas muito destorcidas [Hecht-Nielsen90].

Deste modo, a opção aplicável no problema que se está a tratar passará pelo uso de uma rede neuronal com aprendizagem supervisionada, porque é essa a formulação mais fácil que o problema permite; estabelecer uma lei de transformação entre dois conjuntos de valores concretos — dados sensoriais e valores de ocupação de um mapa. O conjunto de treino consistirá assim nas leituras dos sensores de ultra-som como padrões de entrada, e as correspondentes representações do espaço livre como os padrões de saída.

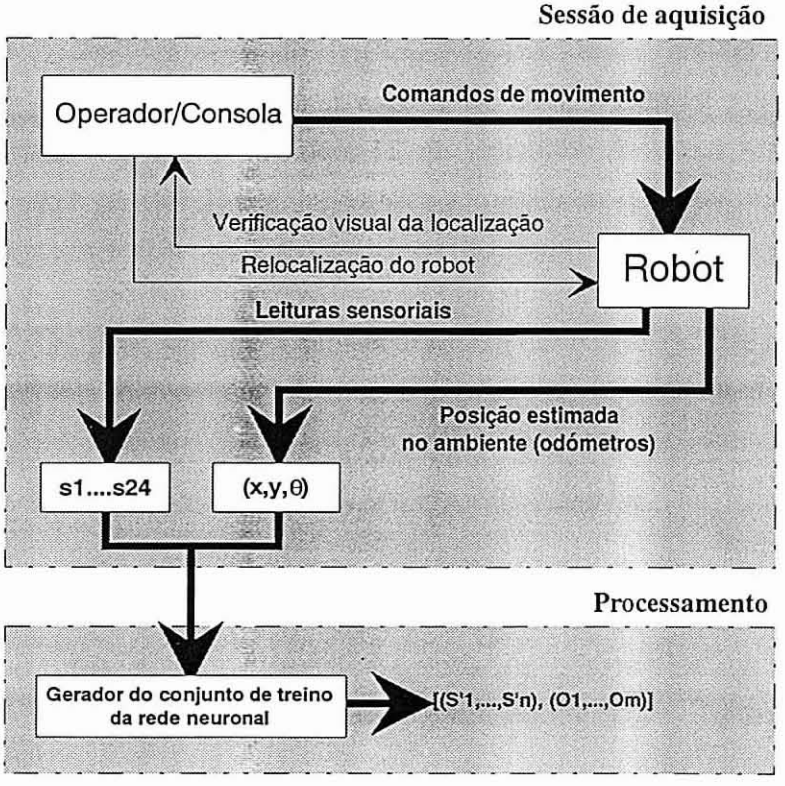


Fig. 31 - Componentes de uma sessão de aquisição: o operador controla o robot e verifica manualmente a sua localização. As leituras sensoriais e a posição estimada são guardadas para posterior utilização.

A aquisição dos dados necessários à elaboração do(s) conjunto(s) de treino tem lugar durantes as **sessões de aquisição**. Estas sessões consistem fundamentalmente em acções de navegação sob o controle do operador. O veículo mover-se-á dentro de um ambiente conhecido e efectuará simultaneamente as leituras sensoriais que serão transmitidas ao operador que as arquivará para futuro processamento. Estas leituras consistem no conjunto de 24 medidas correspondentes a igual número de sensores de ultra-som. Juntamente aos dados sensoriais deverão ser guardadas as coordenadas do robot no momento em que essas leituras foram efectuadas (fig. 31).

2.5.1.4.1. Erros nos dados adquiridos

O conhecimento da posição do robot é de fundamental importância para se poder formular a relação: **dados sensoriais↔espaço livre** para todos os conjuntos de 24 leituras efectuadas durante a sessão. É óbvio que esse conhecimento da posição tem de estar correcto; caso contrário, estar-se-á mais tarde a construir correspondências erróneas entre os dados sensoriais e o espaço realmente livre. Este problema obriga a uma constante contra-verificação da posição e proceder à sua aferição durante toda a sessão.

Verificar manualmente (e continuamente) a posição do robot pode ser bastante trabalhoso, e por certo não isento de erro, mas é indispensável para a geração de um correcto conjunto de treino o mais correcto possível para a rede neuronal.

Outro elemento a ter em conta nestas sessões de aquisição tem a ver com a legitimidade dos dados sensoriais nos seus conjuntos de 24 medidas. Dado que os disparos não são simultâneos, e que as distâncias medidas por cada sensor são diferentes, é pertinente interrogar se as todas as 24 medidas são efectuadas na mesma posição (x,y,θ) do robot. É claro que se o robot estiver em movimento contínuo (como é de prever, e porque será o caso em aplicações reais) as 24 medidas sensoriais não são todas efectuadas na mesma posição do robot. Contudo, é possível estimar alguns erros de medição derivados de uma postura não estática durante as medições.

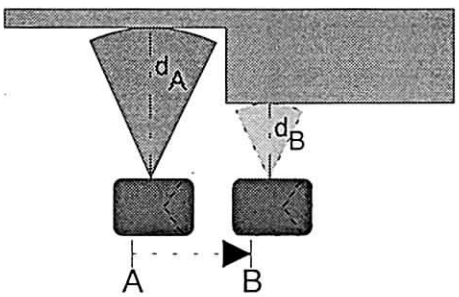


Fig. 32 - Exemplo de erro em leitura de sensor lateral devido ao deslocamento do robot. A leitura do ponto A pode estar disponível apenas quando se chegou ao ponto B.

Os erros mais difíceis de estimar relacionam-se com sensores que efectuam medições em direcções concorrentes (não paralelas) com a direcção do movimento, e sobretudo em situações de rotação do robot. Nestes casos poder-se-á talvez contar com a natureza cónica de um feixe de ultra-som (ver Parte 1). Na verdade, os 20° de abertura de um feixe têm um efeito de integração espacial, que surge aqui pela primeira vez como uma vantagem explícita, senão vejamos: se o robot está a rodar em torno de si próprio com uma velocidade angular de 45° / s (valor correspondente a uma volta completa em 8 segundos) e se se efectuarem 3 medidas por segundo, isso equivale a dizer que o robot varre 45°/3=15° entre amostras, ou seja, um valor ainda coberto por um feixe que apresenta uma abertura de 20°.

Menos fácil de quantificar é o erro resultante de medidas nas direcções concorrentes à do movimento: aqui haverá um forte dependência da distância que se mede no momento, e da variação do próprio ambiente visto do robot. A situação é ilustrada na figura 32: uma medição efectuada quando o robot se encontra no ponto **A** pode estar apenas disponível quando o robot já estiver no ponto **B**, sendo portanto, nesse momento, uma medida errada. No caso ilustrado, a medida é de igual modo perigosa no sentido em que há uma indicação que pode levar a considerar mais espaço livre do que aquele efectivamente existente!

Os motivos de medição tardia (desfasada em relação à posição corrente do robot) podem ser devidos a:

- Distâncias de medição longas (cria atrasos de medição $\approx 2 \times 3$ ms = 6 ms por metro)
- Intervalos de disparo de um sensor (tempo máximo típico usado: 350 ms)

No primeiro caso está inerente o princípio físico do método: a limitada velocidade do som no ar. Medidas longas levam mais tempo a ser efectuadas do que medidas curtas. Todavia, este não será por certo um factor grave se se levar em conta as velocidades esperadas para o robot: por exemplo, uma distância de 5 metros leva pouco mais de 30 ms a ser avaliada, e nesse tempo o robot percorre menos de 2 cm mesmo a uma "grande" velocidade tal como 50 cm/s.

No segundo motivo de medição tardia, trata-se essencialmente de evitar um problema de interferência entre sensores: é necessário garantir que um sensor recebe o seu eco (ou termina o período de "escuta") antes de disparar outro sensor — voltar-se-á a este problema mais tarde. Por conseguinte, um robot deslocando-se a 30 cm/s percorre cerca de 10 cm entre disparos (3 por segundo) de um sensor. É imediata a seguinte importante conclusão, que dita simultaneamente as especificações deste problema: **não se poderão esperar resoluções de medição melhores que 10 cm para o conjunto de parâmetros descritos** (velocidade do

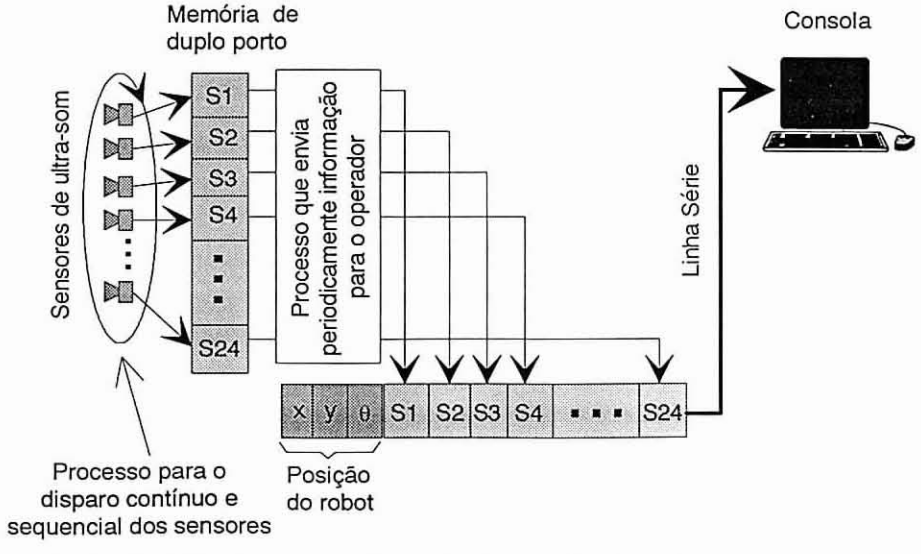


Fig. 33 - Detalhe do processo de aquisição e envio de dados para o operador

robot e cadência da medição de um sensor). A implicação imediata na grelha que serve de suporte aos mapas de percepção é que, nestas condições, não faz sentido ter células de dimensões inferiores a 10 cm.

De todos os modos, é significativo mostrar o esquema de aquisição de dados em maior detalhe para se poder aperceber o nível de **tempo real** dos dados arquivados: na figura 33 pode-se ver que o processo de disparo (e leitura) dos sensores de ultra-som funciona em paralelo com todos os processos restantes e em ciclo fechado. Dessa forma, mantém-se sempre em memória os dados mais recentes no que respeita aos sensores de ultra-som. De igual modo, se possuem sempre os dados mais recentes de **odometria** (x,y,θ). O processo responsável pelo envio de informação para o utilizador, quando chega o seu instante de funcionamento, compila uma cadeia de informação com a posição e as leituras sensoriais e envia-as para a linha série.

2.5.1.4.2. Determinação dos padrões de saída — a ocupação das células.

Terminada a fase de aquisição de dados, dever-se-á passar ao cálculo das ocupações das células a fornecer à rede sob a forma de conjunto de treino, e que é feito, claro, levando em conta a posição supostamente conhecida do robot ao longo da sessão de aquisição. Chega-se assim ao processamento *off line* anunciado anteriormente: determinar quais as células da grelha que estão ocupadas (pelo ambiente e obstáculos) é feito através de uma análise das intersecções da grelha com o mapa do ambiente. O algoritmo completo para o cálculo dessas intersecções está descrito em apêndice, mas a figura 34 pode, para já, servir de ilustração para o problema do cálculo das células ocupadas da grelha.

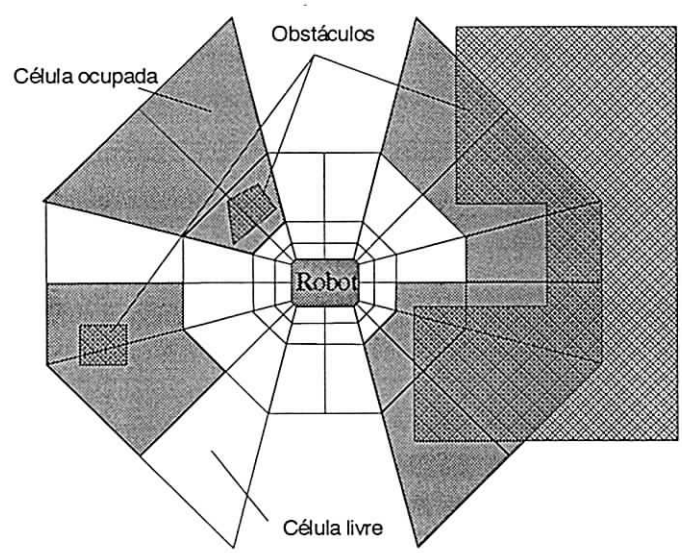


Fig. 34 - Ilustração da determinação de ocupação da grelha por intersecção entre conjuntos de linhas.

Como se pode perceber por uma análise da figura 34, o critério de ocupação de uma célula devido a um obstáculo, ou ao ambiente em geral, não é tão simples como o mero cálculo de intersecção de linhas poligonais. Essa é apenas uma condição suficiente para haver

ocupação, mas não é necessária. Além de outros critérios geométricos (como a inclusão de um obstáculo por parte de uma célula da grelha), é necessário também levar em conta a questão física da percepção: uma célula alinhada radialmente com outra célula ocupada, mas para além desta, é também uma célula ocupada, mesmo se não é intersectada por nenhuma linha do ambiente ou não envolve geometricamente qualquer obstáculo. Está-se de novo a invocar a **continuidade de profundidade** mencionada a propósito das propriedades de uma grelha radial deste tipo.

2.5.1.5. Convergência, erros e parâmetros

Os elementos significativos no processo de aprendizagem de uma rede neuronal relacionam-se com a rede em si e com o próprio algoritmo de retropropagação, e podem listar-se como segue:

- Dimensões da rede.
- Critério de convergência da rede.
- Critério de "reconhecimento" de um padrão.
- Critério de apresentação dos dados no treino (aleatório/sequencial).
- Critério de actualização dos pesos da rede (após cada uma ou todas as amostras de treino).
- Parâmetros específicos do algoritmo (coeficientes de **aprendizagem**, e **inércia** ou **momento**).

As dimensões das redes e os parâmetros do algoritmo serão detalhados e concretizados na secção dos resultados experimentais, dado não haver, à partida, uma lei exacta para os determinar. Dos restantes elementos saliente-se que o critério de apresentação dos padrões de treino foi feito de modo aleatório, e que a actualização dos pesos da rede era feita após a apresentação de cada padrão à rede, e não após cada época (apresentação de todos os padrões de treino). Esse método não gera necessariamente um erro de convergência sempre decrescente (porque corre um risco maior de encontrar mínimos locais do erro), mas a convergência ocorre em menos iterações (portanto, mais rápida).

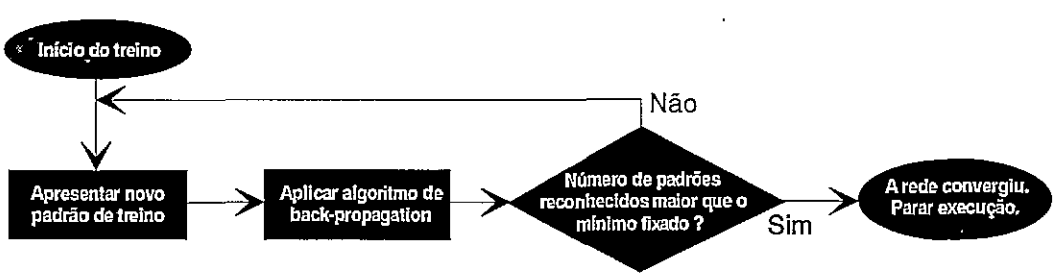


Fig. 35 - Critério de convergência da rede neuronal durante a fase de treino.

O critério de convergência a usar consiste no reconhecimento de um número mínimo de padrões do conjunto de treino (fig. 35). Para monitorar o treino da rede, e obter assim um

primeiro indicador no nível de convergência, pode-se também definir o **erro médio de convergência** dado pela média dos erros em todos os padrões, como se verá adiante.

O reconhecimento ou não de um padrão do conjunto de treino é por conseguinte baseado na avaliação do **erro do padrão.** O cálculo desse erro é baseado no critério da máxima das distâncias de todos os componentes do vector de saída desejado com o vector gerado pela rede, como indica a sequência de expressões que se segue.

$$o_d(k) = \begin{pmatrix} d_0 & d_1 & \cdots & d_{M-1} \end{pmatrix} \tag{9}$$

$$o_t(k) = \begin{pmatrix} t_0 & t_1 & \cdots & t_{M-1} \end{pmatrix} \tag{10}$$

$$\varepsilon(k) = \max_{i=0}^{M-1}\left(\left|d_i(k) - t_i(k)\right|\right) \tag{11}$$

Nas expressões anteriores, $o_d(k)$ é o k-ésimo padrão (vector) de saída desejado e $o_t(k)$ o k-ésimo padrão de saída resultante com o treino até ao momento, e sendo $\varepsilon(k)$ uma distância como indicado, diz-se que o padrão k do total dos T padrões do conjunto de treino é reconhecido se $\varepsilon(k) < \delta$, onde δ é um valor entre 0 e 1 para saídas confinadas no intervalo [0, +1]. Um valor típico para δ é 0.2 ou 0.3, providenciando este último um critério de convergência menos exigente. Uma outra definição de erro de um padrão pode ser dado pela distância Euclidiana entre o padrão desejado e o padrão produzido pela rede — **erro Euclidiano do padrão:**

$$\varepsilon_E(k) = \sqrt{\sum_{j=0}^{M-1}\left(d_j(k) - t_j(k)\right)^2} \tag{12}$$

Estas duas definições de erro de um padrão tem características diferentes: a versão Euclidiana traduz um erro global do padrão, e a versão anterior traduz o erro no mais desfavorável dos casos, mas para esse valor não leva em conta o erro de todos os elementos do vector (padrão). Pode-se dizer que essa definição de erro se adequa mais à classificação de padrões (onde, por exemplo, o elemento mais elevado dos M elementos do vector de saída indica a classe, das M possíveis, do vector de entrada), e o erro com base na distância Euclidiana será por certo mais adequado em redes com o objectivo de interpolar (aproximar) funções contínuas.

2.5.1.6. Uma rede neuronal como sistema interpolador

Quando usadas como interpoladores (aproximadoras de funções), as redes neuronais têm saídas contínuas e também, em geral, confinadas a um dado intervalo. Neste papel, as funções de activação das unidades de saída da rede neuronal são necessariamente funções contínuas (como a função **sigmóide**, ou a **tangente hiperbólica**). Aliás, uma das condições para a aplicação do algoritmo de retropropagação implica uma função de activação continuamente derivável e não constante. Esta condição obriga ao uso de uma dessas tais funções.

Neste problema, podemos fazer corresponder cada célula da grelha a uma saída da rede, e a cada entrada uma leitura de um sensor. Desta forma teremos uma rede com 24 entradas (tantas quantos os sensores) e 60 saídas (60 = 2×6 + 2×8 + 4×8, Cf. fig. 27). Apesar destas saídas contínuas, a rede será ensinada com exemplos onde as saídas são binárias, porque se sabe ao certo quais as células ocupadas e livres: não são definidas células "meio-ocupadas". É claro que na fase de operação, a rede gerará saídas contínuas, mas adoptar-se-á um critério para a binarização dessas saídas. Isto não invalida que não se possam eventualmente usar os valores contínuos como indicadores de quão ocupada uma célula está: depende do sistema que irá interpretar estes valores de ocupação.

A primeira rede a experimentar consiste então em 24 entradas (24 sensores), 60 saídas (60 células) e numa camada de unidades escondidas cujo número de neurões foi sendo adaptado algo empiricamente como se verá mais tarde quando se referirem os resultados experimentais. Nesta rede todas as unidades de uma camada estão ligadas a todas as outras da camada seguinte e/ou anterior. É um caso de ligações totais: isto é, todas as componentes dos dados de entrada estão relacionadas com todas as componentes das saídas. Traduzido em termos práticas, quer dizer que qualquer sensor está em condições de influenciar a ocupação de todas as células da grelha. À partida este facto pode parece excessivo, mas pretendia contemplar eventuais situações de interferência entre sensores, e esperava-se que as ligações (pesos) entre unidades fracamente relacionadas se fosse anulando à medida que a rede ia sendo treinada.

Esta rede forneceu óptimos resultados no sentido em que convergiu eficazmente. Ou seja, o conjunto de treino apresentado, através do algoritmo de retropropagação, permitiu o desenvolvimento adequado de uma função de correspondência entre dados sensoriais e ocupação de células.

Esta convergência foi eficaz (reduzido erro de convergência), mas lenta. De facto, para um conjunto de treino com mais de 3000 padrões de medições/ocupação numa rede de **24::140::60**† necessitou de algumas dezenas de milhões de iterações, ou seja muitos milhares de épocas (uma época equivale a apresentar à rede todo o conjunto de treino uma vez). Computacionalmente isso resultou em vários dias de processamento numa máquina *Sun Sparc 10*!

Como já mencionado, a rede descrita convergiu de uma forma que se mostrará na parte dos resultados experimentais. Porém, ao experimentá-la com dados reais verificou-se a sua ineficácia para um determinado tipo de dados. Testou-se a rede em tempo real, com dados vindos directamente do robot, e constatou-se que as ocupações de células da grelha calculadas pela rede não eram correctas quando o robot se aproximava mais dos obstáculos e ambiente em geral. Enquanto o robot se mantinha a distâncias médias do ambiente, o desempenho da

† A notação **ii::hh::oo** indica uma rede com **ii** unidades de entrada, **hh** unidades na camada escondida, e **oo** unidades de saída.

rede era adequado, mas nas vizinhanças de uma parede, por exemplo, surgiam falhas. A constatação foi que a rede era incapaz de extrapolar. O conjunto de treino parecia então não ser suficientemente representativo; a rede não estava a responder bem para dados completamente alheios aos padrões ensinados. Na verdade, esses padrões tinham resultado de dados reais adquiridos numa sessão de aquisição conduzida manualmente. É claro que, aí, o operador tinha a precaução de não se aproximar demasiado dos obstáculos por questões de segurança. A solução passaria em primeira mão por tentar alargar o conjunto de treino a dados nessas circunstâncias particulares (próximo dos obstáculos), mas isso acabou por se revelar ainda insuficiente. A situação melhorou, mas não a um ponto completamente satisfatório: havia situações que a condução manual não cobria, mas que a rede deveria conhecer na fase de treino para poder generalizar.

Um outro elemento importante são os parâmetros do algoritmo de retropropagação. Não existem regras completamente definidas para os escolher; por isso, foram usados valores dentro dos exemplos mais usuais, como se verá mais tarde nos resultados experimentais.

Dado que se mostrou difícil fazer convergir uma rede global, era de experimentar subdividir o problema da construção do mapa de percepção. Em vez de se definir uma rede única, podem-se definir várias redes menores cada uma responsável por uma determinada zona do mapa, ou seja, responsável por um conjunto de direcções no espaço em torno do robot.

Esta ideia de separar a construção do mapa não verifica uma das vantagens que se tinham anunciado anteriormente: fazer depender a ocupação de cada célula das medidas de todos os sensores. Se isso à primeira vista pode parecer uma vantagem (está-se a prever todo o tipo de interferência mútua e qualquer outra relação entre todos os sensores e todas as células), um segundo olhar indica que se está a sobrecarregar de importância o papel de cada sensor na ocupação geral do espaço. Na verdade, é de esperar que sensores completamente disjuntos na sua área de influência não se afectem mutuamente e, dado cobrirem zonas diferentes, não têm que contribuir para a ocupação das células um do outro. Isto é verdade, porque isso mesmo é procurado aquando do disparo dos sensores, como se verá adiante nos resultados experimentais.

Uma outra questão relacionada com uma rede única, global, a calcular o mapa, é que se está a introduzir um acréscimo elevado de cálculos e ligações sinápticas na rede que são desnecessários uma vez que tenderiam a definir pesos nulos entre alguns neurões (em consequência da inexistência de qualquer relação entre eles). A ideia da independência entre zonas afectas por diferentes sensores é ilustrada na figura 36; remete-se todavia para a figura 25 a descrição mais detalhada da cobertura dentro de uma zona da grelha por parte de cada um dos 3 sensores.

A divisão do mapa, e consequentemente o número de redes a usar, obedeceu em primeira mão à repartição do espaço em torno do robot em 8 zonas, como indicado na figura

20. Cada uma destas zonas é coberta por um grupo de 3 sensores do conjunto dos 24. A correspondência de 3 sensores para cada zona significa que apenas eles determinam a ocupação dessas regiões.

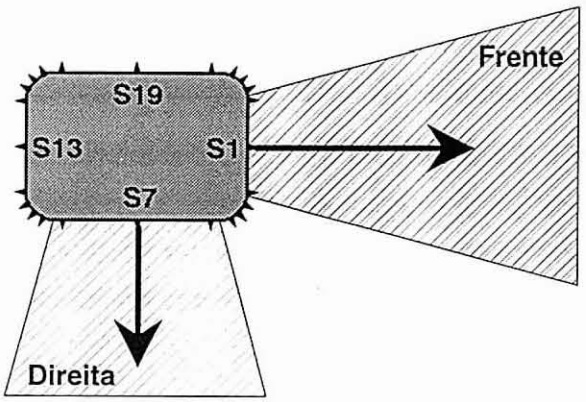


Fig. 36 - Ilustração da independência de zonas da grelha com sensores em direcções perpendiculares.

Esta visão repartida das influências dos sensores é bastante mais simplificada do que a correspondência completa e total na primeira abordagem do problema. O conjunto de treino para estas redes de menores dimensões (**3::h1::6** e **3::h2::8** onde **h1** e **h2** cobriram uma gama variada) são obtidos com subconjuntos de dados do conjunto completo resultante da aquisição. Esta solução não produziu os resultados esperados como se detalhará na parte dos resultados experimentais.

Note-se que foram feitas outras repartições dos sensores pelo espaço em torno do robot: e verificou-se que a convergência tinha tendência a melhorar à medida que aumentava a dimensão do vector de entrada (número de sensores) e consequentemente a dimensão do vector de saída (número de células). A eficácia da rede (quando havia convergência) era sempre semelhante ao caso da rede global: faltava capacidade de extrapolação para alguns dados que eram muito distintos dos dados do conjunto de treino.

Uma possibilidade seria ainda a de ter várias redes parcialmente ligadas umas às outras: desse modo garantir-se-ia uma confinação dos sensores a uma dada região de influência, e salvaguardar-se-iam eventuais dependências (influências de sensores) de uma região para as outras. Porém, dado que as aparentemente verdadeiras limitações do problema (que se verão mais tarde) não prometiam grande sucesso como verificado no caso das redes parciais, e sobretudo devido a algumas limitações do simulador† de redes usado, esta possibilidade de redes parcialmente ligadas (no fundo uma rede global sem algumas ligações entre neurões) não foi explorada.

A justificação desta primeira tentativa de rede neuronal (rede interpoladora), que segue as ideias de redes como aproximadores de funções contínuas (mas treinadas apenas com

† O simulador de redes neuronais é um programa que permite o treino de uma rede com uma dada configuração com um determinado conjunto de parâmetros. O simulador fornece, no fim do treino, e entre outros, os valores das ligações (pesos) entre os diversos neurões. O simulador predominantemente usado neste trabalho foi o ASPIRIN v6.0, que é de domínio público.

exemplos de valores discretos), era a esperança de se poderem vir a utilizar valores contínuos de ocupação em vez do simples "ocupado ou livre". Esse valores poderiam assim vir a ser utilizados para operações de integração temporal ou espacial de mapas, por exemplo. Porém, o sucesso foi limitado, como se referiu atrás e se ilustrará na parte dos resultados experimentais.

2.5.1.7. Uma rede neuronal como sistema classificador de padrões

Após constatadas as dificuldades em encontrar uma rede para gerar a ocupação individual de cada célula, surgiu a ideia de tentar simplificar o trabalho da rede. Como já foi notado anteriormente, o mapa de percepção pode ser dividido em zonas exclusivas de influência dos sensores. Isto significa que um grupo restrito de sensores afecta apenas algumas células do mapa, como parece lógico.

Outra constatação é que o que realmente interessa é determinar se uma célula está ocupada ou não: isto sugere a remoção do carácter contínuo dos valores de ocupação. Desta forma será possível definir padrões binários correspondentes à ocupação do mapa (ou subregiões): valores 1 para células ocupadas e valores 0 para células livres. O que esta ideia propõe é, para a representação do espaço ocupado num mapa, passar de **valores individuais de ocupação de células** para **padrões de ocupação de regiões.**

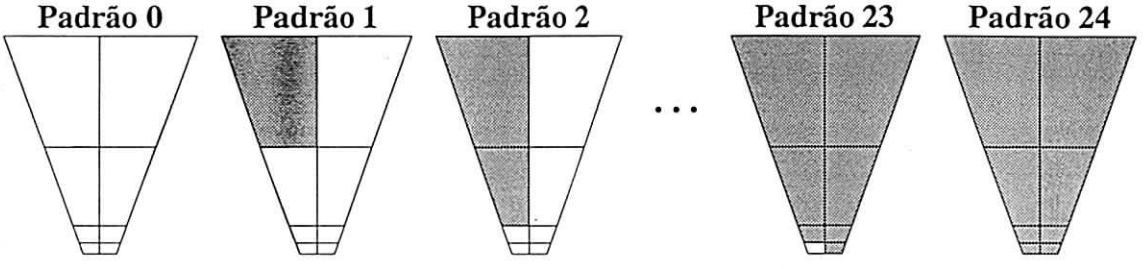


Fig. 37 - Padrões de ocupação de regiões com 8 células. Notar que só há 25 padrões com interesse.

Esta modificação da representação leva a um redimensionamento do problema da construção de mapas de percepção, nomeadamente no que respeita ao número de padrões possíveis de ocupação. Está-se também a migrar para um problema de classificação por oposição aos métodos interpoladores usados primeiro: agora serão redes com uma lei de associação **hetero-associativa** do tipo **vizinho-mais-próximo** (*nearest-neighbour*).

É assim necessária a fragmentação do mapa em regiões para se obter a redução do número total de padrões distintos. Uma forma é seguir a dimensão das regiões que é também imposta e corroborada pela distribuição dos sensores (fig. 37). Sendo agora uma rede puramente classificadora o conjunto de treino deve ser reformatado: os vectores de entrada mantêm-se (os dados sensoriais) mas as saídas deverão ser indicadores de classes, ou seja, consistirá num vector que terá, de cada vez, uma só componente (*bit*) a 1, que será o designativo da classe, ou seja do padrão de ocupação.

Para regiões de 8 células teremos 25 padrões diferentes, e para regiões de 6 teremos 16 padrões. Para regiões com este formato a fórmula geral é: $(n/2+1)^2$, onde n é o número de células da região. Note-se que tentar definir padrões para o mapa global (60 células) seria tarefa impraticável, mesmo quando os aparentes e erróneos 2^{60} ($\approx 10^{18}$) padrões dessem lugar ao real número de $4^2 \times 4^2 \times 5^2 \times 5^2 \times 5^2 \times 5^2 \times 5^2 \times 5^2 = 6.25 \times 10^{10}$ padrões!

Um critério importante para a definição de padrões de ocupação foi o de procurar que, padrões com códigos muito próximos deveriam corresponder a distribuições de ocupação muito semelhantes — isto melhora o treino das redes e a sua capacidade de generalização [pôr aqui uma referência se a encontrar a tempo].

As redes assim definidas obtiveram um grande sucesso. A sua convergência foi muito eficiente (95-99% dos padrões de treino) e a sua capacidade de generalização muito boa, como se verá mais tarde.

A distinção que se tem vindo a fazer entre as noções de **rede interpoladora** e **rede classificadora** é perfeitamente artificial. Conceptualmente trata-se da mesma rede e com o mesmo algoritmo de aprendizagem. Os elementos que mudam duma abordagem para a outra é simplesmente a representação dos dados de saída da rede, logo da dimensão da camada de saída: isso pode trazer outra implicação que é a dimensão da camada interior (escondida) das redes. Na realidade, a rede como classificador não tem necessariamente a tarefa facilitada dado que a correspondência **dados sensoriais↔padrão de ocupação** não é tão directa porque sofre uma transformação um pouco arbitrária introduzida (aliás a experiência indicou que uma rede interpoladora com dados sintetizados e não contraditórios convergia muito bem, como se verá mais tarde). Todavia, uma rede classificadora tem um muito mais restrito universo de possibilidades para a saída, o que em princípio poderá facilitar a capacidade de generalização. Em resumo, uma rede interpoladora, neste problema, tem como saídas valores de ocupação de cada célula, e uma rede classificadora tem como saídas simples indicadores de classes às quais corresponde um número limitado de combinações de células ocupadas e livres. O processo de classificação tem a garantia adicional de gerar apenas valores possíveis no âmbito da lógica do problema; por exemplo, nunca haverá a possibilidade de gerar uma saída onde uma célula próxima do robot esteja ocupada e outra célula alinhada com esta mas mais distante esteja livre, e que se trata de uma situação sem validade real.

2.5.1.8. Os dados para o conjunto de treino

Dada a inconsistência dos dados de treino verificada na prática e que se mencionará na parte dos resultados experimentais, em simultâneo à reformulação do problema das redes como classificadores, decidiu-se filtrar os dados de todas as medições contraditórias devido às reflexões especulares: ou seja, em termos práticos forneceram-se os dados quase ideais (sem contradições), mas mantiveram-se os casos não contraditórios que preservam mesmo assim as reflexões especulares, e juntaram-se também dados sintéticos para o restante da gama de

distâncias, ou seja aqueles que não podem ser adquiridos experimentalmente por limitações práticas de navegação.

Para uma melhor convergência da rede neuronal, e para uma melhor actuação na sua fase de operação, o conjunto de treino deve ser o mais equilibrado possível. Isto quer dizer que as classes que a rede deverá vir a reconhecer devem estar representadas o mais equitativamente possível. Os dados experimentais não facilitam esta tarefa, e este foi mais um motivo porque se utilizaram dados sintetizados, uma vez que correspondem a situações não verificáveis na prática. Este tipo de procedimento é frequente quando se pretende uma rede neuronal para lidar com sistemas experimentais; basta pensar por exemplo no caso já referenciado de D. Pomerleau [Pomerleau93] onde situações impossíveis de atingir experimentalmente foram geradas (computacionalmente) para completar melhor o conjunto de treino da rede neuronal.

2.5.2. Outros Métodos para calcular os Mapas

O método baseado nas redes neuronais tem como base as razões apresentadas relativamente á capacidade das redes de combinar informação redundante mas eventualmente intrincada e difícil de analisar por métodos tradicionais. Das redes espera-se também uma certa imunidade a alguma informação errada por ser completamente desconhecida, mas também a capacidade de reconhecer semelhanças com situações apresentadas na fase de treino e poder gerar as saídas adequadas. Faz contudo sentido tentar outros processos para a construção dos mapas e efectuar comparações.

2.5.2.1. O método da intersecção com o polígono do espaço livre

O método da **Intersecção com o Polígono de Espaço Livre (método IPEL)**, é muito

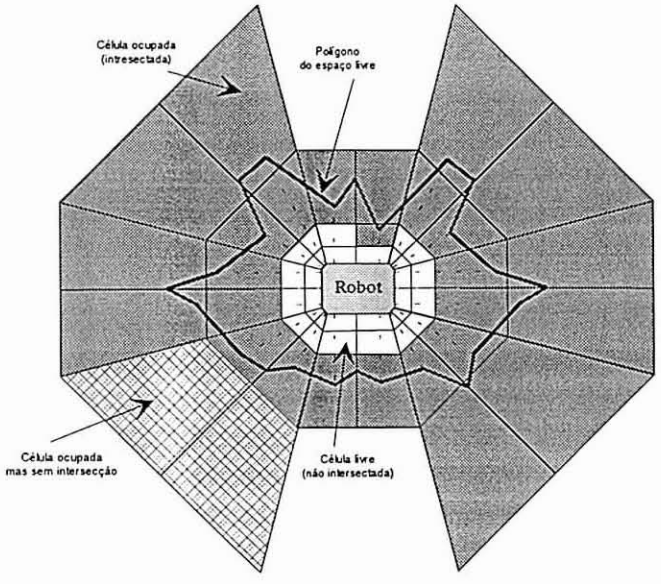


Fig. 38 - Ilustração do cálculo do mapa de percepção pelo método da intersecção com o polígono do espaço livre (IPEL)

elegante pela sua simplicidade funcional mas, como verificaremos, muito pesado computacionalmente.

O método IPEL consiste muito simplesmente em definir o polígono cujos vértices são dados pelas medidas sensoriais em torno do robot. Em seguida calculam-se as intersecções com as células da grelha: onde houver intersecção, há ocupação. Depois, o algoritmo invoca as propriedades deste tipo de grelhas e faz um varrimento de todas as células para etiquetar como ocupadas as células para além de uma célula ocupada.

Na prática, o algoritmo vai verificar a intersecção do polígono de espaço livre com cada célula da grelha. Uma variante computacionalmente mais favorável começa a verificar as células mais próximas do robot em primeiro lugar. O método encontra-se ilustrado na figura 38.

Além da simplicidade conceptual, o método IPEL apresenta uma vantagem que resulta também da própria natureza da grelha, que é orientada para este tipo de sensores: trata-se de uma certa continuidade de ocupação. Quando por exemplo, uma medida é muito mais curta que as adjacentes, o polígono resultante apresenta uma concavidade notória, e as células naquela direcção e em uma ou duas direcções adjacentes acusam a ocupação resultante. Tem-se assim casos de sensores a influenciar regiões que não apenas a sua (fig. 39).

A primeira grande desvantagem do método IPEL é o seu custo computacional: trata-se de testar a intersecção de um polígono com 24 lados (para este número de sensores) com toda uma série de polígonos correspondentes às células da grelha (60 no caso mais geral usado). Uma versão modificada testaria apenas os segmentos de recta que constituem as células uma só vez, reduzindo na prática para cerca de metade o número de polígonos celulares a testar, mas mesmo assim o custo computacional pode não ser suportado pelo sistema do robot.

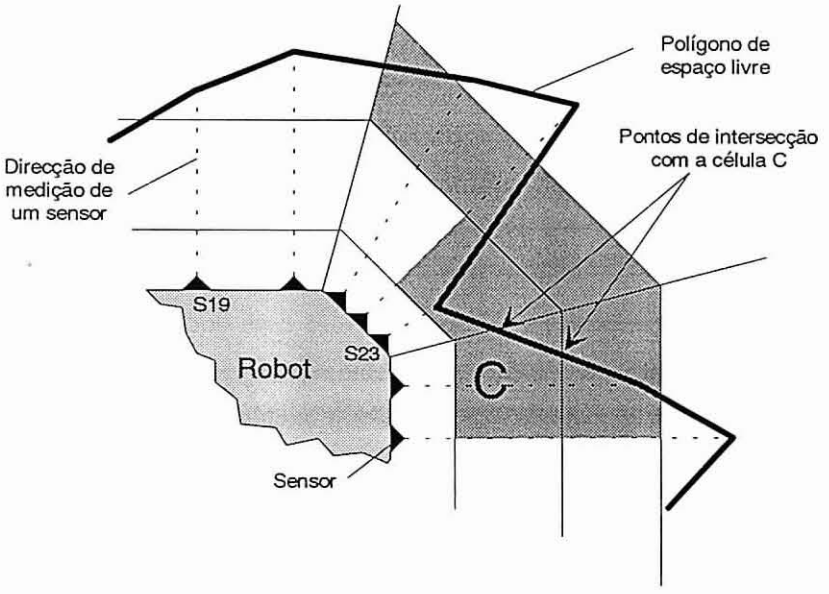


Fig. 39 - Uma vantagem do método IPEL é garantir uma certa continuidade geométrica no mapa de percepção resultante: aqui a célula C é dada como ocupada devido à medida curta de S23, que "pertence" a outra região do mapa que não a região da célula C.

A segunda desvantagem é que os mapas resultantes variam à taxa de actualização dos sensores, e um sensor num ponto de indecisão (medida especular intermitente) alterna em mapas consecutivos possivelmente bastante distintos, o que impede uma representação estável do espaço livre em torno do robot. Se inclusivamente um sensor falha (deixa de funcionar) o polígono torna-se altamente convexo e gera um mapa excessivamente ocupado impedindo o seguimento da navegação nas direcções afectadas. Ou seja, não permite muito bem contar com a redundância de sensores.

A terceira desvantagem é, ironicamente, consequência da vantagem em demasia da continuidade de ocupação. De facto, verificou-se experimentalmente que os mapas resultantes eram sobreocupados: isto é, havia uma tendência para desperdiçar espaço livre. Este fenómeno talvez pudesse ser minorado com modificações da geometria (dimensões) das células, mas isso estaria fora de questão dada a escolha mais ou menos cuidada destas dimensões celulares.

2.5.2.2. Um método baseado em regras de comparação

Uma segunda alternativa, que bem poderia ser a primeira por ser mais intuitiva, é definir um conjunto de regras de interpretação das medidas sensoriais e construir o mapa com base nelas. As regras usadas podem-se resumir no seguinte:

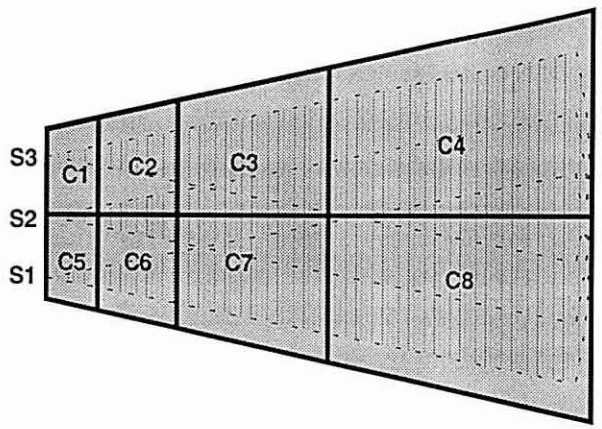
- Cada região é afectada (coberta) por 3 sensores apenas (os "seus" sensores).
- O sensor mais à direita (vendo para fora do robot) afecta apenas as células da direita da região.
- O sensor mais à esquerda afecta apenas as células da esquerda da região.
- O sensor do meio afecta todas as células da região.
- Definem-se gamas de distância de acordo com a extensão das diferentes células.
- A medida mais curta dos pares de sensores (S3,S2) e (S2,S1) (fig. 40) define a célula ocupada mais próxima do robot.

Este método não lida explicitamente com as situações de reflexão especular, mas de certa forma pode colmatar algumas situações de um sensor faltoso porque a decisão é feita com base em dois sensores, e é a distância mais curta que prevalece.

Um método destes é demasiado simples para lidar com dados faltosos, e a redundância espacial dos sensores é só indirectamente aproveitada (o facto de S2 poder decidir para as células todas). É de esperar à partida que seja um sistema que reaja imediatamente às flutuações sensoriais, como também já mencionado no método anterior (IPEL).

Na prática este método deu resultados algo mais interessantes que o método IPEL, mas tal como previsto os mapas sucessivos podiam diferir muito devido às tais medições indecisas ou oscilantes de sensores em situações eminentes de reflexão especular. Deste modo também aqui não havia uma representação estabilizada do espaço, sobretudo com o veículo em movimento. Os mapas conseguiram mesmo assim ser usados pelos algoritmos de navegação

(como se verá adiante) mas o comportamento era bastante mais irregular do que quando se usaram mapas gerados por uma rede neuronal.



$$C_k = \begin{cases} f(d_k, \min(S_3, S_2)) \text{ para } k = 1,2,3,4 \\ f(d_k, \min(S_2, S_1)) \text{ para } k = 5,6,7,8 \end{cases}$$

C_k é o valor de ocupação das respectivas células na região da grelha (ao lado), e S_1, S_2 e S_3 as medidas apresentadas pelos respectivos sensores. f() é a função geradora das ocupações (livre/ocupado) tendo em conta as dimensões d_k das células.

Fig. 40 - Dependência das ocupações das células com as medidas sensoriais num método baseado em regras simples de análise dessas medidas.

2.5.2.3. Breve comparação dos métodos

Dos três métodos propostos e experimentados, o baseado numa rede neuronal provou ser aquele que simultaneamente lidava melhor com as reflexões especulares e que gerava mapas mais estáveis. Isto quer dizer que mapas sucessivos não apresentavam as flutuações (de ocupação de certas células) dos outros dois métodos. Uma forma de verificar claramente esta situação era em posição estática quando havia um ou mais sensores não adjacentes em oscilação por efeito de reflexão especular.

A forma natural como foi concebido o método de criação de mapas com as redes neuronais permite facilmente introduzir extensões, como por exemplo acrescentar a cada conjunto de 3 sensores os dois sensores adjacentes. Desse modo forma-se assim uma rede com 5 entradas, mas preservando a saída tal como foi definida. Pretende esse facto refinar uma certa componente de continuidade geométrica do ambiente: esperar-se-á que a rede consiga utilizar esta informação complementar para melhor concluir sobre os resultados das medições dos sensores "exteriores" (S1 e S3 na figura 40) de cada região.

2.6. Possibilidade de introdução de memória recente no sistema

2.6.1. Mapas de percepção instantânea e integração de mapas sucessivos

Os mapas de percepção gerados, tal com descrito até aqui, são mapas de percepção instantâneos. Isto significa que traduzem leituras sensoriais feitas num dado momento, independentemente de leituras anteriores. Como vimos, este carácter não é perfeitamente instantâneo se o robot se estiver a mover, porque aí, e pelas razões enumeradas atrás, as leituras vão sendo feitas à medida que ele se move. Embora se possam estimar os erros de

medição que advêm do movimento do robot, não seria talvez de desprezar uma tentativa de integração de mapas consecutivos, e assim minorar eventuais flutuações de medidas.

A representação de mapa usada é, já em si, uma forma de integração sensorial (múltiplos sensores para uma mesma região), e o uso de redes neuronais para o construir é ainda uma fase posterior de integração, pelas propriedades inerentes aos sistemas neuronais enunciados anteriormente. Está-se a falar, claro está, em integração espacial. A corroborar estes preceitos verificou-se o facto que a variabilidade dos dados sensoriais tem repercussões minimizadas nos mapas gerados.

A integração temporal (como a média de ocupações de mapas consecutivos) poderia suavizar ainda mais a variação do espaço livre detectado, mas implica logo à partida uma definição das especificações da dinâmica do robot: para manter legítima a pretensão de integrar mapas no tempo ter-se-ia que garantir que os mapas corresponderiam à mesma posição do robot no espaço, ou à mesma posição salvo um dado erro. Isto seria simplesmente definido pelo espaço que o robot percorre entre mapas, ou seja, obrigaria a levar em conta a taxa de criação de mapas e a velocidade do robot.

Em alternativa à integração temporal, que só faz sentido em situações estáticas ou quase-estáticas, resta ainda uma certa possibilidade de integração espacial no próprio ambiente. Este problema é bastante mais complicado, porque levanta de imediato a questão do que é que é pretendido com essa integração espacial no ambiente! Poder-se-ia querer construir uma representação da ocupação do ambiente em torno do robot com degradação de resolução espacial mas com mais certezas de valores de ocupação. Este facto traduz-se pela ideia de construir um outro tipo de mapa à custa destes mapas de percepção. Esse novo mapa seria igualmente solidário com o sistema de coordenadas do robot mas seria calculado em função dos últimos N mapas de percepção e da variação de posição do robot entre cada um.

Surgem de imediato algumas questões: qual o tipo de geometria ou topologia deste novo mapa **integrado no espaço**? Seria uma mapa de células regulares ou não? Se sim, qual a sua forma e quais as suas dimensões, e qual a legitimidade de gerar um mapa regular a partir de mapas não regulares? Se fosse um mapa de células não regulares, como seria a sua distribuição/orientação? Temos aqui um problema que é fundamentalmente topológico mas também prático. Não serão já os próprios mapas de percepção suficientemente adequados e enquadrados às características do problema (natureza dos sensores e fiabilidade sensorial, redundância, ...)? Por certo sim. Foram concebidos para obedecer às propriedades do sistema completo, e por isso, a sua topologia apareceu como a mais adequada relegando para outro plano outras topologias. Assim, dadas as inúmeras questões surgidas relativamente à possibilidade da integração espacial e tendo a consciência que os mapas de percepção em si só já oferecem um certo grau de integração, foi decidido não progredir mais nessa direcção sem primeiro testar a eficiência e fiabilidade dos mapas tal como foram definidos.

2.6.2. A possibilidade de uma rede neuronal dinâmica

Uma outra forma de introduzir memória no sistema de construção de mapas de percepção pode ser feita logo a nível sensorial. A memória traduz-se aqui por integração temporal. A ideia seria a de usar uma rede neuronal que tenha como entradas, os valores correntes das medições, bem como alguns valores anteriores desses mesmos sensores: estamos na presença das chamadas TDNN (*Temporal Delay Neural Networks*) — Redes Neuronais com Atraso Temporal ou Redes Neuronais Dinâmicas.

Também com este processo haveria que impôr restrições na dinâmica do robot em função da taxa de dados sensoriais disponível: para os fins a que se destina (avaliar o espaço livre num dado instante), é ilógico tentar combinar (relacionar) dados sensoriais oriundos de diferentes posições no espaço, o que sucederia realmente se a velocidade do robot fosse tal que as distâncias que se desloca entre amostras serem superiores ao erro máximo admitido para essas mesmas amostras! Este efeito poderia ser colmatado se a rede neuronal dinâmica, a par das entradas de dados no momento actual e nos momentos anteriores, tivesse a velocidade do robot (isto é, do mapa de percepção que é lhe solidário) como uma outra entrada. Desse modo, seria, em princípio, possível esperar que a rede usasse os mapas anteriores na elaboração de mapas de percepção mais robustos. Uma variante seria ainda usar (também como entradas da rede) os mapas calculados anteriormente em vez dos dados que lhe deram origem. Nesta última abordagem, a parte mais difícil (ou mais trabalhosa) seria obter o conjunto de treino para a fase de aprendizagem da rede neuronal dinâmica.

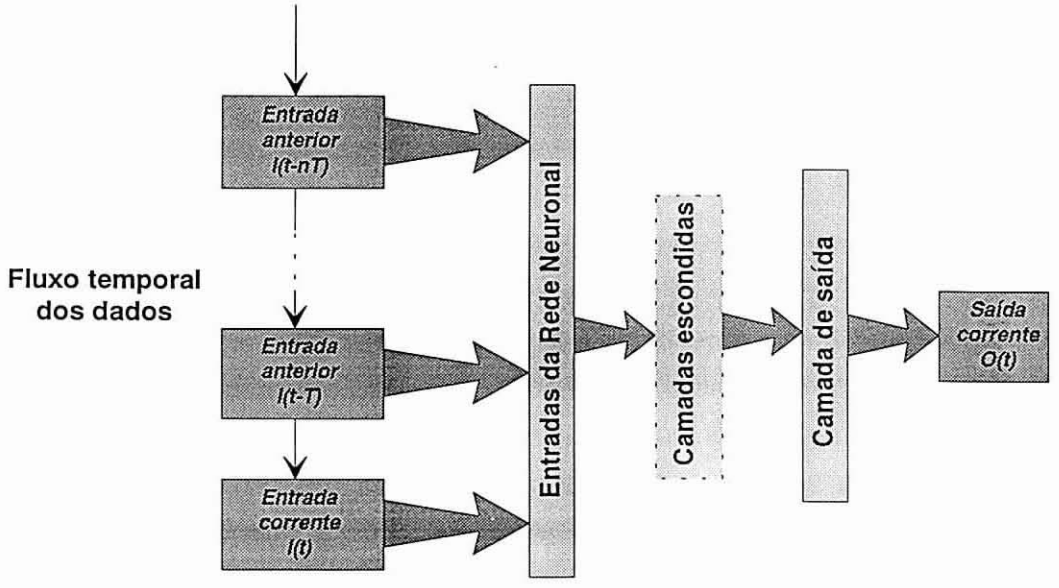


Fig. 41 - Ilustração do princípio de uma TDNN

2.7. Navegação baseada em mapas de percepção

A navegação com base nos mapas de percepção é um processo que envolve a tomada de decisões de movimento face ao espaço livre e ocupado que os mapas traduzem. Um bloco que

se designará por Navegador Local será o responsável pela leitura dos mapas e com base no que à frente se definirá estratégia de navegação gerará movimento que se repercutirá em novas posições do robot, e consequentemente, alteração do espaço ao seu redor. Isso implicará diferentes leituras sensoriais, logo novos mapas, e novos comandos de movimento. Estes conceitos elementares na navegação local estão ilustrados na figura 42.

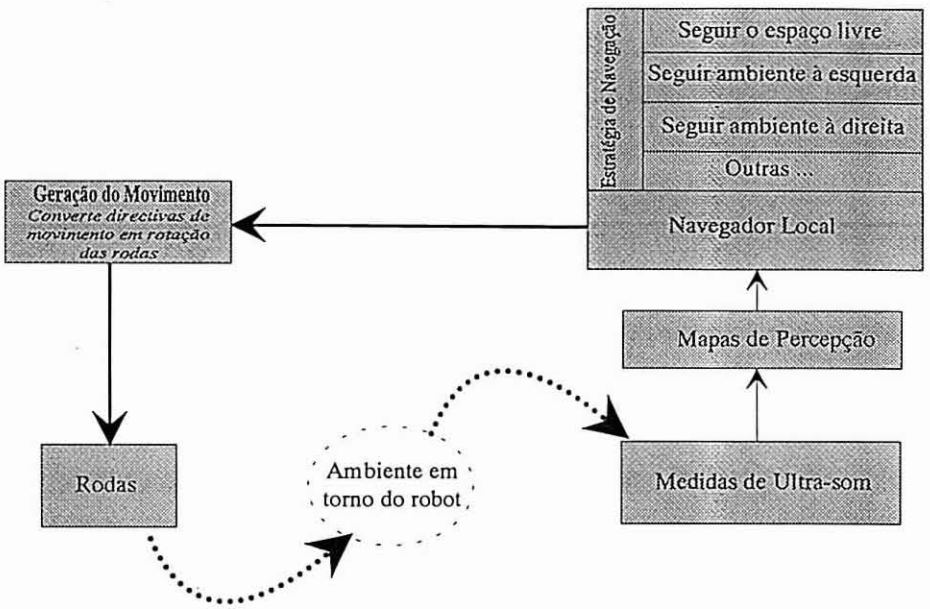


Fig. 42 - O processo da navegação local. O ciclo inerente a esta tarefa fecha-se pelo ambiente.

2.7.1. Estratégias de navegação local

Conforme foi definido, a Navegação Local consiste na apreciação do espaço livre em torno do robot, e na execução de movimento dentro desse espaço livre captado pelo sistema de percepção. A execução de movimento é feita segundo um conjunto de interesses ou directivas impostos por um outro sistema a mais alto nível conceptual (como se verá adiante). Definir-se-á então **Estratégia de Navegação Local** (ou apenas estratégia de navegação) como o conjunto de acções necessárias para seguir e/ou manter um determinado **comportamento** de movimento do robot. Uma estratégia de navegação comporta de certa forma a resolução automática dos problemas holonómicos: gerará a **nova direcção** e **intensidade** do movimento apenas! Fala-se aqui de direcção e intensidade do movimento, o que supõe logo de início um controlo do movimento feito em velocidade. A alternativa seria efectuar um controle de movimento por posição, isto é, por deslocamentos do robot. A questão é que esse tipo de controlo vai um pouco contra as características de incerteza de posição que envolvem todo este processo da navegação local, o que por si só seria já uma razão suficiente para secundar, neste contexto, o movimento controlado por posição. Por outro lado, o controlo por posição resultaria num movimento por certo muito irregular dado que seria necessário, em cada ponto, calcular valores adequados de velocidade (e aceleração) para a execução de cada troço de deslocamento. No controle por velocidade, a continuidade do movimento é mais facilitada: a

velocidade é a grandeza directa a usar, e as acelerações para as mudanças de velocidade esperadas (isto é, as admitidas pelo sistema) não são muito relevantes, podendo contar-se com os valores de defeito do sistema do robot. Desta forma, fica justificada a opção pelo controle em velocidade e, consequentemente, as grandezas a manipular: **direcção** e **intensidade** do movimento (velocidade).

Um comportamento é, essencialmente, uma tendência geral de reacção ao espaço em torno do veículo. Essa tendência pode ser a de manter as maiores distâncias possíveis ao ambiente, ou inversamente, tentar manter um flanco do robot a uma dada gama de distâncias desse ambiente.

2.7.1.1. As estratégias possíveis

Definida a ideia do que é um comportamento é imediato precisar estratégias. Rapidamente se pode conceber que existem duas atitudes perante o ambiente: procurar o espaço livre, ou procurar o espaço ocupado. A primeira entende-se como o desejo de se manter afastado o mais possível dos obstáculos e corpos do ambiente em geral. A segunda, é mais delicada e pode ser vista como o mover-se "amparado" com a presença de uma referência que se não quererá perder, ou seja procurar manter-se vizinho ao próprio ambiente e fazer dele a referência de navegação (situação de contorno de um obstáculo). Tem-se assim que as duas linhas possíveis para estratégias são **seguir o espaço livre** e **seguir o ambiente.** Esta última com duas variantes, que envolverá os flancos direito ou esquerdo do próprio robot.

Saliente-se que não é legítimo (ou pelo menos útil) pensar em estratégias que procurem uma direcção instantânea fixa, como por exemplo: "mover-se tentando manter uma orientação que faz 45° com a orientação neste momento". Este tipo de procedimento contraria a filosofia da definição de estratégia de navegação por exigir uma saída (acção) quantitativa. Ora isso é inviável com a resolução da representação espacial que se possui, nomeadamente por uma certa escassez de dados e pela própria natureza dos sensores usados (ultra-som). Por outro lado obrigaria ainda a um seguimento (*tracking*) das orientações do robot para poder concluir quanto à orientação instantânea relativamente à orientação inicial (isto é, no momento da selecção da estratégia).

Uma outra ideia poderia ser a de estratégias associadas a uma maior ou menor força (rapidez) em procurar a direita ou a esquerda; deste modo não seriam mais do que variantes das estratégias de seguir o ambiente já mencionadas, apenas mudaria o tempo que levavam a chegar-se à esquerda ou à direita. Todavia seria difícil a cada momento escolher qual o grau de atracção para um dos flancos. Tudo isto é mais um problema de afinação de parâmetros do que a introdução de novas estratégias.

Um elemento que eventualmente poderá fazer parte de uma estratégia, e que estará associado a uma espécie de pré-execução de um comportamento, será uma qualquer espécie de condição inicial. Essa condição inicial poderá ser, por exemplo, uma orientação inicial com

pura componente de rotação em malha aberta, isto é, baseado exclusivamente no sistema de odometria do robot. Note-se porém, que se trata assim duma definição já bastante lata de estratégia de navegação.

Finalmente, uma última ideia para nova estratégia é a de **comutação dinâmica de estratégia.** Seria uma estratégia que consistiria em dar a liberdade ao Navegador Local de escolher uma das outras estratégias de acordo com a distribuição do espaço livre (poder-se-ía fixar a nova estratégia nesse mesmo momento ou permitir que se continuasse a ajustar dinamicamente). Trata-se assim mais de uma **hiper-estratégia** ou **meta-estratégia,** visto não definir em si qualquer comportamento, mas permitir a comutação de estratégia entre as outras de seguir o espaço livre ou de seguir o ambiente à esquerda ou à direita. Esta estratégia da comutação dinâmica poderia ser do tipo: "passa a seguir o ambiente à esquerda se o espaço livre à esquerda é menor do que à direita, e vice-versa". Deste modo, o robot entraria automaticamente numa operação de contornar (talvez indefinidamente) obstáculos sempre que se aproximasse de um deles.

2.7.1.2. Outras particularidades dos mapas de percepção baseados em GGG

As grelhas que tem sido usadas até aqui são grelhas completas que servem para representar um mapa do ambiente em torno do robot em **todas** as direcções. Ora, e dado que o veículo não se desloca simultaneamente para trás e para a frente, e para reduzir os custos computacionais dos mapas, é de todo vantajoso trabalhar com uma grelha reduzida. Compare-se uma destas grelhas (fig. 43) com a grelha geral (fig. 27), e verifica-se que, das 8 regiões trapezoidais, foram suprimidas as 3 da retaguarda. Quando o veículo estiver a andar para trás, considerar-se-á tão simplesmente uma grelha simétrica desta (sem parte frontal, mas com as partes da retaguarda).

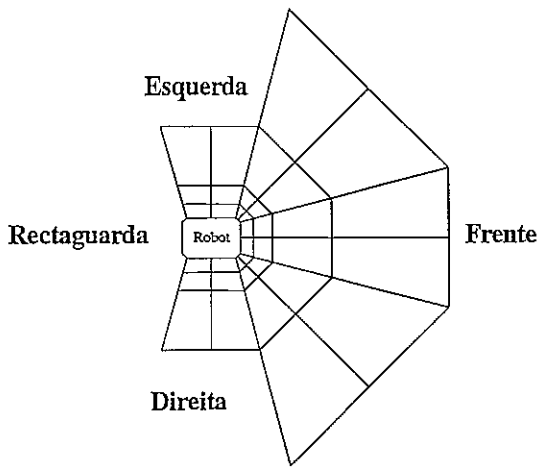


Fig. 43 - Grelha reduzida que permite mapas menores, logo uma maior taxa temporal de mapas.

Relembre-se que esta redução (simplificação) da grelha tem como único fundamento melhorar as prestações computacionais dos algoritmos de cálculo dos mapas: um sistema sem

essas limitações pode perfeitamente manter uma grelha global para ter sempre um mapa global do espaço em torno do robot.

Dentro destas GGG podemos definir, agora por outras conveniências como se verá mais tarde no desenvolvimento dos algoritmos de navegação, outros agrupamentos de células além das regiões trapezoidais: são os **sectores de ocupação** e as **camadas de proximidade** (fig. 44).

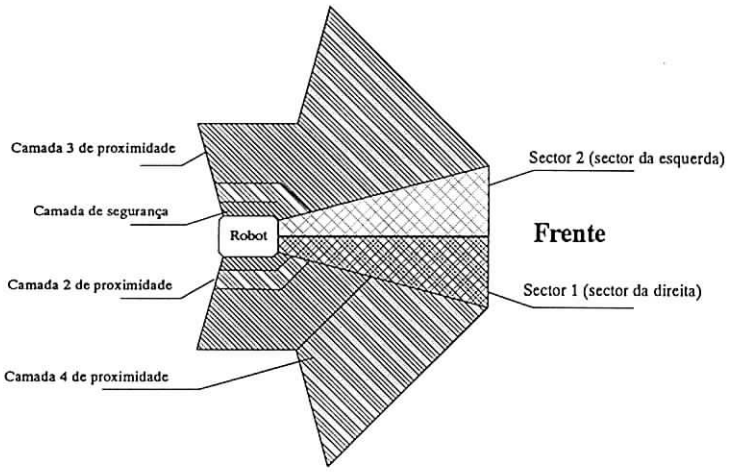


Fig. 44 - Camadas de proximidade e sectores de ocupação de uma grelha.

2.7.1.3. Implementação algorítmica das estratégias de navegação

A implementação das estratégias consiste então em desenvolver métodos computacionais para o cálculo das componentes do movimento preconizado pela estratégia, que são a **nova direcção** (orientação) e a **intensidade da velocidade** nessa direcção. As gamas de valores em condições normais (isto é, na possibilidade de movimento) são [-90°,+90°] para a direcção. A velocidade, em teoria, admite qualquer valor, mas contingências de implementação neste sistema impõem os valores {-10, -9, ...,-1, 0, +1, ...,+9, +10} para a velocidade de cada roda, onde os valores representam aproximadamente dm/s.

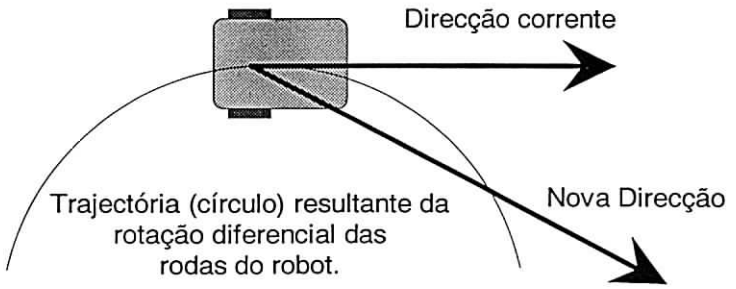


Fig. 45 - A Nova Direcção **do robot: o seu cálculo é feito (nas estratégias de navegação) em função dos mapas de percepção.**

Do ponto de vista de dinâmica há uma série de grandezas com que se pode (e deve) lidar, e que surgem no conjunto das equações gerais do movimento: **velocidade** e **aceleração**

linear, e **velocidade** e **aceleração angular**. É óbvio que as grandezas angulares são as responsáveis pela mudança de direcção.

A nova direcção (isto é, a direcção a procurar, e determinada segundo os preceitos da estratégia), será uma direcção medida instantaneamente com a direcção corrente do robot, ou seja, a tangente à trajectória (fig. 45). A nova direcção tem "validade" até um novo valor ser calculado (de acordo com, eventualmente, novas condições de ocupação de espaço em torno do robot). Deste modo, se uma **nova direcção** desejada não for nula (0°), o robot descreverá uma trajectória perfeitamente circular até um novo valor de **direcção** ser imposto.

A intensidade da velocidade não é tão significativa nos propósitos da estratégia de navegação quando comparada com a direcção: pretende manifestar a maior ou menor celeridade do robot na execução do movimento na **nova direcção**. As indicações para o seu cálculo serão especificadas abaixo.

Ficou assim ilustrado um método algorítmico para a determinação das grandezas do movimento preconizado pelas estratégias, e que se detalha a seguir. Todavia, outros métodos (não algorítmicos) poderão ser experimentados, e no final desta secção procuram-se ideias para isso mesmo.

2.7.1.3.1. Cálculo do movimento: a nova direcção

Neste processo algorítmico, a (nova) direcção desejada **D** para a orientação do robot é calculada como uma combinação vectorial (Eq. 13) das chamadas **Direcções Fundamentais**. Essa combinação tenderá a gerar uma direcção onde a ocupação é mínima, e leva em conta a distribuição de ocupação de células em cada direcção.

$$\mathbf{D} = \sum_i \alpha_i \Psi_i(C_i)\mathbf{V}_i, \quad i = esquerda,\ direita,\ frente,\ semi\text{-}esquerda,\ semi\text{-}direita \tag{13}$$

α_i *Parâmetros especiais dependentes da estratégia*
Ψ_i *Função escalar de ponderação de células*
C_i *vectores com a ocupação das células na direcção fundamental*
$\mathbf{V}_i$ *Versor para cada direcção fundamental*

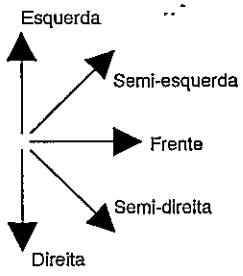


Fig. 46 - As cinco Direcções Fundamentais.

As direcções fundamentais são um conjunto de cinco direcções especiais, vistas do robot, e que estão associadas a uma certa repartição do espaço (fig. 46). Podem também ser direcções de movimento do robot às quais correspondam combinações pré-definidas do

movimento das rodas; por exemplo, a direcção **Frente** corresponderá a ambas as rodas com igual velocidade, e as direcções **Esquerda** e **Direita** corresponderão aos casos em que as rodas têm velocidades exactamente opostas (simétricas). Note-se que as direcções não são linearmente independentes, mas isso é irrelevante: elas pretendem é traduzir direcções essenciais de movimento, não definir uma base de um espaço vectorial.

É mais ou menos claro que cada tipo de estratégia de navegação requererá diferentes conjuntos de parâmetros para o cálculo da nova direcção. Por exemplo, um **seguidor de espaço livre** (que se definirá melhor mais tarde) terá parâmetros perfeitamente simétricos: os da esquerda iguais aos da direita na expressão (13). Por outro lado, as estratégias **seguidoras do ambiente** (também explicadas adiante) terão assimetrias claras exactamente nesses mesmos parâmetros. Na prática, parâmetros nulos ou negativos deverão ser evitados para permitir que situações de bloqueio possam ser mais facilmente resolvidas.

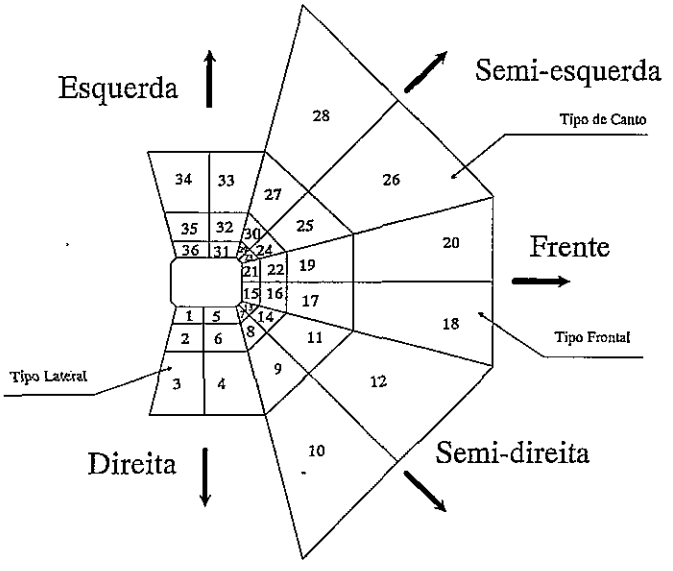


Fig. 47 - As direcções fundamentais e os tipos de regiões em torno do robot (frontal, lateral e de canto). A numeração no interior das células é meramente indicativa e consequência da definição da GGG.

2.7.1.3.1.1. Cálculo da função de ocupação Ψ

Continuando a analisar a expressão (13) tem-se que, para cada direcção, a avaliação da importância (ponderação) das células livres é executada por uma função específica, que se designa por Ψ, cujo argumento é o vector com a ocupação das células, e que tomará em conta as propriedades individuais de cada uma; por exemplo, células mais próximas terão uma importância maior que as outras. A definição das várias funções de ocupação será feita com base na direcção fundamental a que dizem respeito. No caso corrente, as **Direcções Fundamentais** caracterizam regiões (também designadas um tanto abusivamente por cones) de três tipos diferentes: **frontal, de canto,** e **lateral**. Para essas regiões ou cones (que se identificam, aliás, com os já definidos trapezóides da GGG), a função Ψ é muito semelhante: a única diferença ocorre entre os tipos **frontal** ou **de canto,** e o tipo **lateral**, onde o vector de

ocupação tem um número diferente de componentes — 6 em vez de 8. A função Ψ tem assim um valor que exprime a ocupação de uma dada região (cone). Esse valor é calculado com base numa contabilização de células livres, mas atribuindo pesos diferentes às células consoante a sua localização no cone: na figura 48 sugerem-se simples exemplos de coeficientes de ponderação para os três tipos de regiões em causa.

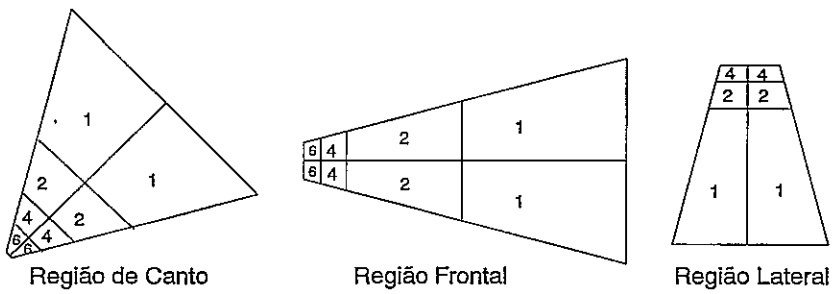


Fig. 48 - Ilustração de valores para os parâmetros de ponderação da ocupação das células dos diferentes tipos de região.

É de esperar que os coeficientes de ponderação dependam da posição da célula na região e das suas próprias dimensões, o que, relembrando razões expostas anteriormente, traduz de certo modo o grau de certeza da ocupação dessas células. É também claro que, numericamente, o valor de Ψ deverá ser normalizado para uniformizar os pesos das regiões no cálculo final da nova direcção desejada. A expressão para a função de ocupação para uma dada região (e direcção fundamental associada) vem então da seguinte forma:

$$\Psi(C) = \frac{\sum_{i=0}^{N} \gamma_i (1 - c_i)}{\sum_{i=0}^{N} \gamma_i} \quad , \; N = 6 \; ou \; 8 \qquad (14)$$

C é o vector de ocupação de uma região
c_i são os valores de ocupação (0 →livre, ou 1→ocupada)
γ_i são os coeficientes de ponderação para o tipo de região

Se todos os c_i (elementos de C) são nulos, ou seja, não há ocupação, a função tem valor 1; isto significa que essa direcção fundamental terá um peso máximo na determinação da nova direcção para o robot. Se por outro lado todos os c_i são iguais a 1 (ocupação total nessa região), o valor da função resultaria em 0, e essa direcção fundamental não terá peso algum na determinação da nova direcção. Porém, essa situação é desaconselhável: a função nunca deverá ter um valor nulo para possibilitar a saída de eventuais situações de bloqueio. Para esses casos, a função deverá retornar um valor muito pequeno (por exemplo, ≈ 0.01) mas não nulo. Tem-se então para a função de ocupação de uma região o seguinte:

$$0 < \Psi(C) \leq 1 \quad , \quad C = \left(c_1, \ldots, c_{6\,ou\,8}\right) \qquad (15)$$

2.7.1.3.1.2. Os parâmetros das estratégias (α_i)

Para o cálculo da direcção final são tomadas em conta as ocupações das diferentes direcções, e os seus valores são aferidos com um conjunto de parâmetros [os α_i na expressão (13)]. São portanto, também estes, parâmetros ponderadores, mas agora de direcções fundamentais. Os seus valores dependem da estratégia de navegação em uso. Como visto anteriormente, existem fundamentalmente duas estratégias:

1. Seguir o espaço livre.
2. Seguir o ambiente a um flanco (esquerda ou direita)

Seguir o espaço livre significa procurar a direcção mais favorável a um maior afastamento do ambiente e obstáculos. Conceptualmente traduz-se por procurar as direcções "menos ocupadas". **Seguir o ambiente a um flanco** significa deslocar-se procurando manter uma distância de segurança (e não menos) do ambiente que vá surgindo pelo flanco em causa. Estas estratégias de seguir o ambiente têm componentes atractoras e repulsoras, contrariamente à de seguir o espaço livre que só tem componentes repulsoras.

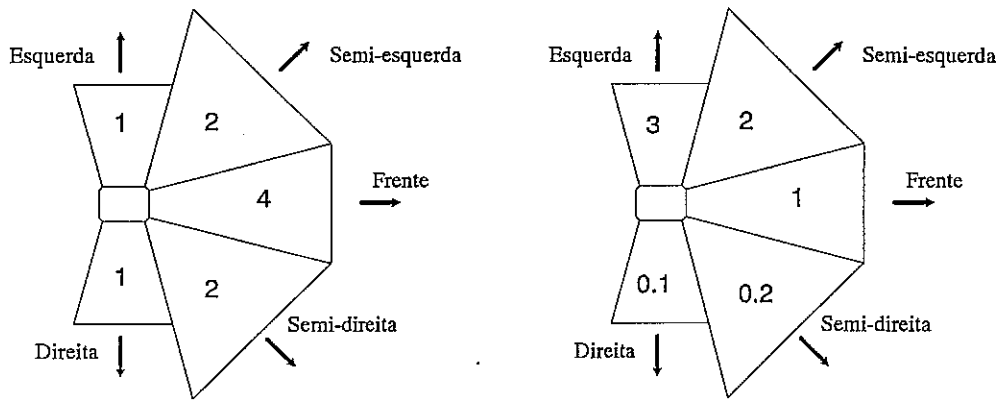


Fig. 49 - Ilustração dos parâmetros das estratégias de navegação para o cálculo da nova direcção. a) Um seguidor de espaço livre. b) Um seguidor do ambiente à esquerda.

É bem claro, e já foi dito, que uma estratégia para seguir o espaço livre, pondera de igual modo (ou pelo menos de modo simétrico em relação ao eixo longitudinal do robot) todas as direcções do espaço. Analogamente, seguir o ambiente à direita ou à esquerda, significa desequilibrar a importância das ocupações nas várias direcções fundamentais e gerar por conseguinte uma tendência "atractora" para um desses flancos.

A figura 49 mostra (de forma meramente ilustrativa) as duas situações básicas:

2.7.1.3.1.3. As componentes finais da direcção

Como foi dito atrás, e nesta abordagem em particular, as direcções fundamentais são em número de cinco e como tal (vectores no plano) não são linearmente independentes. Convirá deste modo reduzir a expressão final para uma representação em duas direcções linearmente

independentes; desse modo torna-se mais fácil a manipulação do vector com a direcção final. Suponha-se então quer as duas direcções independentes são escolhidas como sendo a **Frente** e a **Esquerda.** Os versores (vectores unitários) das outras direcções são definidos pelas seguintes expressões:

$$\mathbf{V}_{Direita} = -\mathbf{V}_{Esquerda} \tag{16}$$

$$\mathbf{V}_{Semi-esquerda} = \frac{\sqrt{2}}{2}\mathbf{V}_{Frente} + \frac{\sqrt{2}}{2}\mathbf{V}_{Esquerda} \tag{17}$$

$$\mathbf{V}_{Semi-direita} = \frac{\sqrt{2}}{2}\mathbf{V}_{Frente} - \frac{\sqrt{2}}{2}\mathbf{V}_{Esquerda} \tag{18}$$

Deste modo, obter-se-ão de forma expandida as seguintes componentes para o vector nova direcção, que se designa por **Vector da Direcção da Mínima Ocupação — VDMO** *(Directional Minimum Occupancy Vector —DMOV)*:

$$\mathbf{VDMO} = FRENTE \cdot \mathbf{V}_{Frente} + ESQUERDA \cdot \mathbf{V}_{Esquerda} \tag{19}$$

$$ESQUERDA = \left(\alpha_E \Psi_E + \sqrt{2}\alpha_{Se}\Psi_{Se}\right) - \left(\alpha_D \Psi_D + \sqrt{2}\alpha_{Sd}\Psi_{Sd}\right) \tag{20}$$

$$FRENTE = \alpha_F \Psi_F + \sqrt{2}\alpha_{Se}\Psi_{Se} + \sqrt{2}\alpha_{Sd}\Psi_{Sd} \tag{21}$$

Sendo F = Frente, E = Esquerda, Se = Semi-esquerda, Sd = Semi-direita, e
$\Psi_i \equiv \Psi_i(C_i)$, $i \in \{Esquerda,\ Direita,\ Semi-esquerda,\ Semi-direita\}$

$C_i = (c_1, c_2, \cdots, c_n)$, $n = \begin{cases} 8 \Leftarrow i \in \{Frente,\ Semi-esquerda,\ Semi-direita\} \\ 6 \Leftarrow i \in \{Esquerda, Direita\} \end{cases}$

Tem-se finalmente, que a nova direcção obtém-se rodando o veículo o seguinte valor:

$$\Delta\theta = \arctan\left(\frac{ESQUERDA}{FRENTE}\right) \tag{22}$$

Da expressão anterior, rapidamente se constata que se a frente estiver ocupada (FRENTE $\approx$ 0) a estratégia de navegação tentará reorientar o veículo impondo um ângulo grande (±90°).

2.7.1.3.1.4. As situações especiais no cálculo da direcção

A fórmula geral para o cálculo da nova direcção permite lidar com uma grande parte das situações teóricas de ocupação de espaço em torno do veículo, mas há uma série de situações que carecem de tratamento especial, isto é, independentemente do valor da nova direcção calculada.

A questão de fundo é que uma mera contabilidade de células pode ser insuficiente para uma determinação adequada da nova direcção: é necessário levar em conta a distribuição de células livres (nomeadamente em sectores de ocupação). Isso pode ser feito mais facilmente

com uma análise de distribuição do que procurar seleccionar minuciosamente coeficientes de ponderação celular que traduzam correctamente esses casos.

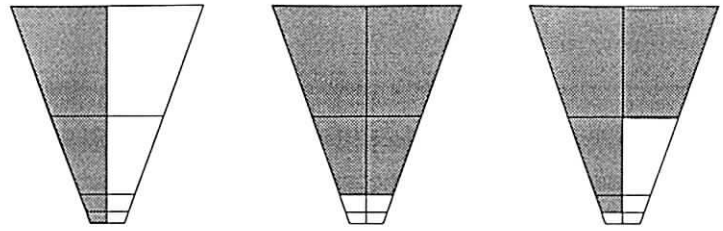


Fig. 50 - Várias possibilidades de ter uma região com 4 células ocupadas. A mera contabilidade de células pode ser insuficiente para calcular a nova direcção.

Veja-se uma situação de, por exemplo, 4 células ocupadas na região frontal (fig. 50): há várias possibilidades (combinações), e duas delas implicam não se poder avançar mais para a frente – o caminho está de certeza impedido! Como distingui-las sem ambiguidade apenas com coeficientes de ponderação? É claro que este exemplo de problema não existiria se as regiões associadas às direcções fundamentais consistissem apenas de células alinhadas radialmente; contudo, essa situação exigiria o dobro das direcções fundamentais do que aquelas definidas aqui, o que não é desejável, nem prático (direcções menos interessantes), nem necessário na maioria das situações (estar-se-ia a tender para o caso de ter tantas direcções fundamentais quantas os sensores no sistema, que é a situação que se quer obviar por gerar interpretação errónea das medições)

É também verdade que a análise das **camadas de proximidade** (fig. 44) pode ser um complemento à direcção determinada algoritmicamente: dessa análise se poderão detectar situações que uma parametrização em coeficientes de ponderação (gerando um valor escalar) não permite expressar facilmente.

Outro caso de situação especial é quando a passagem está perfeitamente obstruída, seja para ir em frente ou para rodar para um dos flancos (beco sem saída). Lidar com essa situação pode tornar-se complicado se se quiser evitar ciclos oscilatórios de movimento: há que evitar que o robot entre em oscilação com a introdução de histerese no processo de determinação na nova direcção a seguir.

Estas e outras situações especiais (não contempladas portanto pela fórmula geral do cálculo da nova direcção) serão mais detalhadas e ilustradas na descrição do algoritmo final para as estratégias de navegação, feita na parte dos resultados experimentais.

2.7.1.3.2. Cálculo do movimento: a nova velocidade

Sendo a nova direcção do robot o elemento fundamental na geração de movimento para uma estratégia de navegação local, resta definir a velocidade do robot para se completar a execução do movimento. Como já referido atrás, esta velocidade pretende ser a velocidade linear do robot. Relativamente à velocidade calculada pela estratégia de navegação, duas premissas básicas se podem impor desde já:

- Deve ser proporcional à quantidade de espaço livre (na regiões frontais, essencialmente)
- Nas rotações deve ser especialmente limitada por questões de segurança (baixa rotação diferencial das rodas)

A primeira premissa "sugere" que se utilize o valor de **FRENTE** na expressão (21) como um primeiro indicador para a velocidade. Isso é legítimo uma vez que essa indicação exprime a resultante do espaço livre nas direcções frontais. Deste modo, o espaço completamente livre corresponderá à velocidade máxima permitida.

Dadas estas indicações, para passar ao cálculo da velocidade cada uma das rodas atente-se à figura 51. Nela está representado o robot com as rodas e as indicações das velocidades interessantes no sistema. Em termos de dinâmica, o robot pode ser reduzido (simplificado) a três pontos rigidamente ligados: os pontos correspondentes aos extremos do eixo tractor (X_L e X_R) e o ponto **A** que contém a massa do sistema.

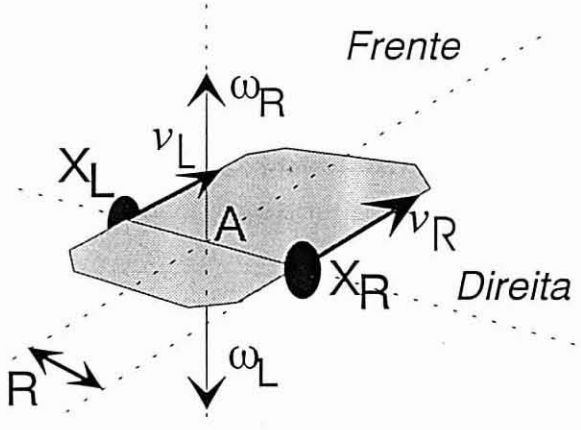


Fig. 51 - A velocidade angular de um ponto do robot em função das velocidades lineares das rodas.

É sabido que a velocidade angular de um ponto **X** qualquer do robot em relação a um ponto de referência **A** é dada pela razão entre a velocidade linear relativa desse ponto **X** e a distância que os separa: $\overline{AX}$. Supondo que o ponto **A** tem velocidade v_A, a roda da esquerda tem velocidade v_L e a da direita velocidade v_R, e que a resultante da velocidade angular do ponto **A** (desprezando seu o sentido final) é dada pela adição das velocidades angulares dos pontos X_L e X_R, tem-se o seguinte:

$$\bar{\omega}_A = \bar{\omega} = \bar{\omega}_R + \bar{\omega}_L \qquad (23)$$

$$\omega_R = \frac{v_R - v_A}{R} \quad \text{e} \quad \omega_L = -\frac{v_L - v_A}{R} \qquad (24)$$

$$\omega = \frac{1}{R}(v_R - v_L) \qquad (25)$$

Rescrevendo a expressão anterior para as unidades de grau por segundo (°/s) sabendo que a correspondência entre grandezas medidas em radianos e em graus sexagesimais é dada por:

$$\omega = \theta \cdot \frac{\pi}{180} \tag{26}$$

Onde ω *vem em radianos e* θ *vem em graus sexagesimais*

Daqui tem-se que a velocidade angular do robot em função do valor das velocidades das rodas é dada por:

$$\theta = \frac{180}{\pi R}(v_R - v_L) \qquad [°/s] \tag{27}$$

Se se quiser impôr ao veículo uma variação de direcção θ [°], mas a uma cadência de N vezes segundo (por ser esta a cadência à qual se determinam as novas direcções desejadas), tem-se que a velocidade angular a impôr ao robot é $N \cdot \Delta\theta$ [°/s]. Assim ter-se-á finalmente:

$$N \cdot \Delta\theta = \frac{180}{\pi R}(v_R - v_L) \tag{28}$$

Desta expressão resulta uma infinidade de combinações de velocidades para a roda esquerda e para a roda direita do robot de forma a que ele tenha uma dada velocidade angular. Assim, é necessária uma segunda condição para o cálculo concreto (numérico) para as velocidades das rodas. Uma condição *ad-hoc* é que nenhuma das velocidades (esquerda ou direita) supere os valores limites ou de segurança: assim pode-se sugerir como segunda condição o seguinte:

$$v_R + v_L = 2 \cdot v = 2 \cdot v_A \tag{29}$$

Esta expressão resulta numa proposição trivial quando as velocidades das rodas são iguais (nesse caso a velocidade do ponto **A** do robot é igual à velocidade de ambas as rodas). Sabendo que v_A é imposto de acordo com as condições listadas anteriormente, tem-se finalmente para o cálculo da velocidade das rodas do robot a resolução do seguinte sistema de equações:

$$\begin{cases} v_R - v_L = \dfrac{N \cdot \Delta\theta \cdot \pi \cdot R}{180} \\ v_R + v_L = 2 \cdot v_A \end{cases} \tag{30}$$

N - frequência do processo de imposição da nova direcção (em Hz)
R - semi-distância entre rodas (em metros)
$\Delta\theta$ *- variação de orientação a desejada (em graus)*
v_A *- velocidade linear instantânea máxima a que o robot se pode mover (em m/s)*

2.7.1.4. Outras possibilidades para a implementação de estratégias de navegação

Dado que uma estratégia consiste essencialmente num problema de determinar grandezas de movimento (direcção e velocidade) obedecendo a uma série de parâmetros tais

como a distribuição do espaço livre, é natural que se possam encontrar métodos (e até princípios) alternativos.

Um princípio alternativo ao da "contabilização" ponderada do espaço livre em zonas seria o de formular o problema em termos de determinar, a cada momento, a posição ideal do robot (na região de espaço livre) de forma a minimizar um conjunto determinado de parâmetros, como por exemplo a distância média da fronteira do robot aos limites dessa mesma região (fig. 52).

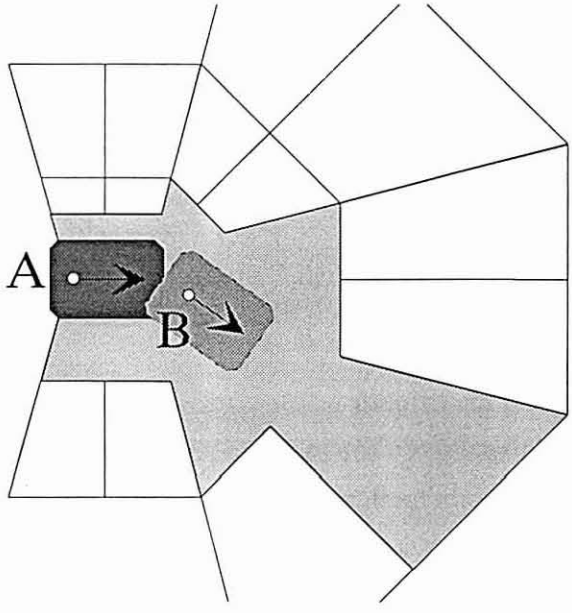


Fig. 52 - Possibilidade para um princípio alternativo de estratégia de navegação: cálculo do reposicionamento do veículo na região de espaço livre com maximização de distâncias entre polígonos.

Este método careceria ainda do cálculo dos elementos dinâmicos do robot: isto é, velocidades e acelerações para se deslocar do ponto A para o ponto B na figura 52. Um controlo aparentemente por posição poderia eventualmente dar lugar a um controlo por velocidade e manter a tal continuidade e "fluência" do movimento.

2.7.1.4.1. Possibilidade de redes neuronais para cálculo de estratégias

Como descrito atrás, o processo de base para as estratégias de navegação é fundamentalmente algorítmico. Todavia, inclui também alguns elementos de análise de premissas para as situações especiais em que o cálculo puro não permite lidar sem complexidade. A natureza aparentemente complexa deste problema pode sugerir abordagens diferentes para implementar as estratégias de navegação como entidades que determinam qual o movimento a impor com base no espaço livre que lhes é comunicado.

Um sistema neuronal poderia ser uma possibilidade: por exemplo, em primeiro lugar, um operador humano executaria acções de navegação, e de seguida, fornecer-se-iam os dados do espaço livre (mapas de percepção) e dos movimentos do robot a uma rede neuronal, que aprenderia a "conduzir" como o operador humano. É claro que se supõe que a navegação

humana era simples e fundamentalmente coerente. Mais uma vez se antevê a vantagem de um controlo de movimento por velocidade.

2.8. Uma arquitectura completa de navegação

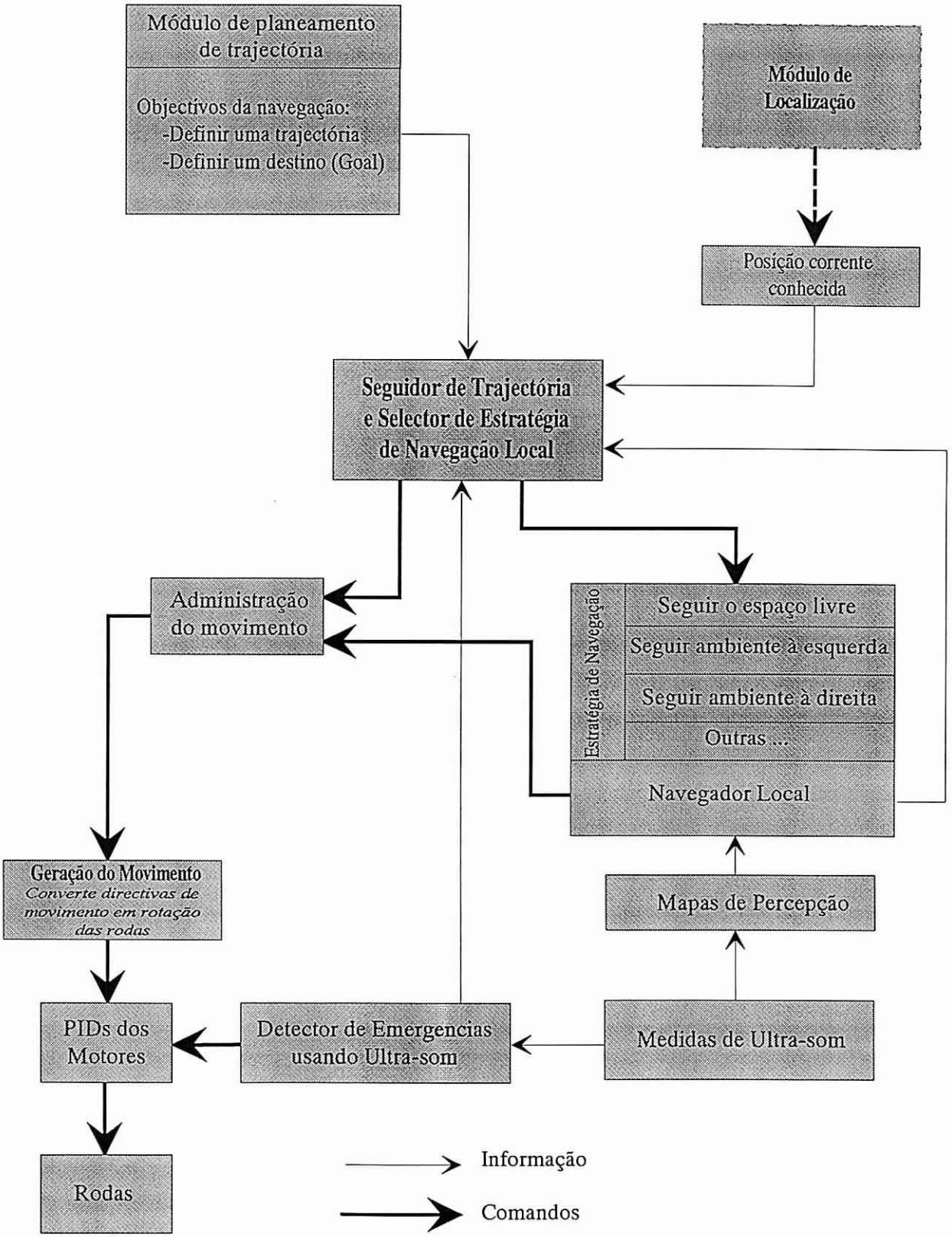


Fig. 53 - Arquitectura completa de navegação, incluindo módulos auxiliares.

A navegação local é apenas uma componente num sistema completo de navegação. Uma arquitectura completa é proposta na figura 53. Aí poderemos ver um conjunto de blocos responsáveis cada um por uma série de problemas específicos, e quiçá, independentes uns dos outros.

Esta arquitectura difere de algumas arquitecturas tradicionais sobretudo pela sua extrema modularidade, independência de processos e número de **ciclos fechados de navegação**.

A arquitectura proposta surge na linha das arquitecturas de *subsumption* [Brooks86], que se caracterizam pela existência de camadas modulares de acção entre os sensores (percepção) e os actuadores do robot (acção). Opõe-se à organização tradicional que corresponde à existência dos procedimentos e análises ("raciocínios", no fundo) entre os sensores e actuadores, e que, segundo Brooks, se pode representar como na figura 54.

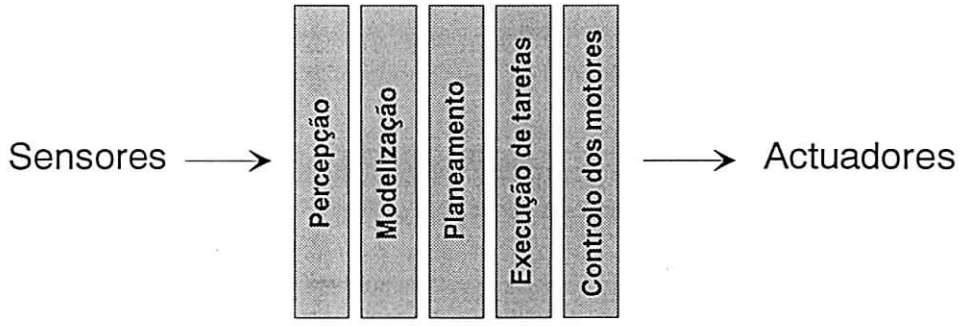


Fig. 54 - Decomposição tradicional do sistema de controlo de um robot móvel em módulos funcionais.

A alternativa proposta por Brooks está esquematizada na figura 55 e corresponde a uma decomposição bastante geral do problema global da robótica móvel. Consiste em múltiplas camadas (níveis) obedecendo a uma organização hierárquica a tal ponto que as camadas superiores podem inibir (*subsume*) as saídas das camadas inferiores.

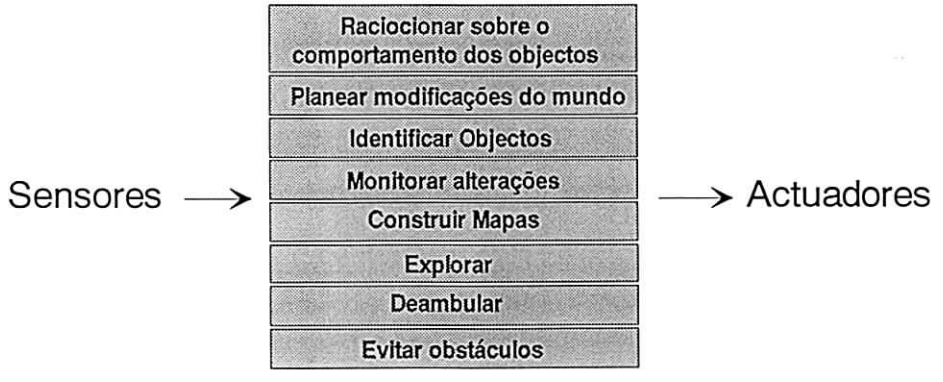


Fig. 55 - Decomposição do sistema de controlo de um robot móvel

Na arquitectura aqui proposta são também de realçar os ciclos fechados de navegação: estes ciclos ocorrem a três níveis. O mais baixo nível começa nas leituras sensoriais que são função da posição no ambiente, que é função do movimento das rodas do robot — este ciclo

fecha-se com o detector de emergências que actua sobre as rodas quando há eminência de colisão. Um segundo ciclo fechado funciona a médio nível e as suas componentes são:

posição → dados → mapas → navegador local → movimento → rodas → posição

O terceiro ciclo fechado ocorre a um nível mais conceptual: o ciclo é similar ao de médio nível, mas o fluxo de informação ao longo dele passa pelo selector de estratégias em vez de pelo navegador local: ou seja é o selector de estratégia, juntamente com o administrador do movimento que se impõe ao movimento sugerido pelo navegador local.

Estas características ajudam a enquadrar a arquitectura proposta na categoria das arquitecturas reactivas, que Arkin [Arkin94] define do seguinte modo:

> [...]O controlo reactivo é uma abordagem da robótica que elimina o uso de representações e "raciocínios" intermédios durante a execução da missão de um robot. Existe assim um acoplamento forte entre a percepção e as acções dos motores. Esta estratégia permite respostas em tempo real e é particularmente bem adaptada para ambientes dinâmicos ou não modelados[...]

É precisamente a existência dos referidos ciclos, pelo menos os de mais baixo nível, que permite o tal acoplamento entre a percepção e a acção.

Uma questão fundamental, em especial na implementação onde todo este trabalho foi aplicado, relaciona-se com a localização de cada módulo no sistema *hardware*. Dado ter começado como um sistema perfeitamente manual, grande parte do controlo residia numa estação remota, ficando no robot apenas o sistema de interpretação dos comandos manuais. Com o desenvolvimento das arquitecturas (de navegação e de *software*) o sistema no robot começou a tornar-se complexo e, por isso, a separar-se da estação remota. A primeira grande razão para esse facto foi a limitada largura de banda do canal de comunicação entre a estação remota e o robot. De seguida, foram as razões de índole autonómica que fomentaram essa migração: quanto mais do sistema estiver no robot mais se aproximará da autonomia.

No estado até onde este trabalho foi levado, no robot residiam todos os módulos da arquitectura representada na figura 53 excepto o módulo de **Localização**, e o módulo de **Planeamento de Trajectória** que, tal como o módulo **Seguidor de Trajectórias** ainda não tinham sido implementados (ver apêndice **Descrição das infra-estruturas e ambiente de trabalho**).

2.8.1. O papel de cada módulo da arquitectura

• Planeador de trajectória

Este módulo está associado aos mais altos níveis do controlo da navegação. Terá o conhecimento (mais ou menos completo) do ambiente onde trabalhar, e conhecerá os

objectivos das tarefas. Está em condições de fornecer uma primeira aproximação dos pontos de passagem, mas sem indicar como fazê-lo: trata-se de um "**fazer o quê?**" em vez de "**como fazê-lo?**" (fig. 56). O planeador de trajectórias está em condições de alterar os pontos previstos de passagem sempre que o ache necessário.

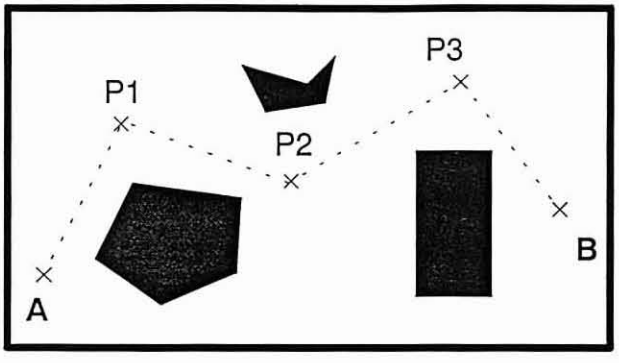


Fig. 56 - O Planeador de trajectórias, tendo conhecimento do ambiente, pode definir toda uma trajectória a executar, ou simplesmente pontos de passagem (de A para B passa-se por P1, P2 e P3).

• Estimador da posição corrente (localização)

Conforme se constatou na Parte 1 deste texto, uma tarefa completa de navegação em robótica móvel só é possível se existir a capacidade de efectuar a localização do robot dentro do ambiente, que se supõe mais ou menos conhecido. Excluem-se os casos de mera deambulação no espaço livre, e casos de construção de mapas de ambiente, que são também tarefas de navegação mas de aplicação talvez um pouco mais restritas, como por exemplo explorações planetárias e dos fundos marinhos.

Desta forma, estimar a posição corrente através de observação do ambiente é necessário pelo menos por uma razão: verificar se se chegou ao destino previsto! Outros pontos intermédios de localização podem ser necessários, isto dependerá das características (qualidade) da trajectória prevista pelo módulo de planeamento, mas dependerá também das diferenças entre o ambiente supostamente conhecido e o ambiente real, ou seja os obstáculos desconhecidos. A tarefa de localização pode ser demorada, e aí talvez seja necessário reduzir o número de vezes em que se fará localização. Por outro lado, é possível um dia chegar a um sistema que faça localização em tempo real. A estrutura da arquitectura e os seus módulos conseguem lidar com situações onde o ambiente possa ter sido alterado ou mal descrito, e onde o evitar dos obstáculos desconhecidos continue a ser um preceito importante, o que parece indiscutível neste momento.

• Seguidor de trajectória e selector da estratégia de navegação local

Este módulo terá um papel muito complexo. Será ele a traduzir os objectivos de navegação e/ou pontos de passagem, em orientações de navegação para os módulos de mais baixo nível. Verificar se uma trajectória (ou a linha que une pontos de passagem) está a ser cumprida terá de ser feito com o auxílio de um módulo de localização. Portanto, seguir uma dada trajectória (mais ou menos grosseiramente) pode consistir em fornecer toda uma

combinação de estratégias de navegação que leve o robot até ao seu objectivo. As estratégias de navegação, em geral, estão definidas de forma a evitar os obstáculos que o sistema de percepção encontre. Desta forma, entregar o controle de velocidade e direcção a um módulo de navegação local contém intrinsecamente toda a segurança no movimento, e no destino a seguir; este último, pelo menos a curto prazo, ou seja, até se poder fazer uma nova localização. A função de selecção de estratégias traduz em resumo as acções deste módulo: não haverá uma combinação única de sequências de estratégias que levem o robot dum ponto ao outro, cabe a este módulo de determinar as melhores ou mais desejáveis combinações.

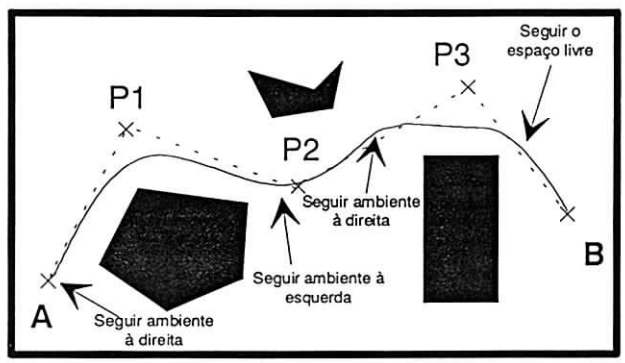


Fig. 57 - Exemplo de sequência de estratégias que o seguidor de trajectórias poderia impôr ao módulo de navegação local para ir de A a B. (as setas indicam o momento de início da nova estratégia)

• Administração do movimento

Este módulo tem uma função quase de *multiplexer*. A estrutura modular desta arquitectura exige um bloco deste tipo a partir do momento em que mais do que um módulo podem gerar ou produzir "intenções de movimento". Neste caso temos o navegador local, que seguindo as directivas que lhe são fornecidas do "alto", usa os mapas de percepção e gera de forma contínua uma nova intenção de movimento (nova velocidade e direcção). Paralelamente poderá surgir a necessidade de impôr uma velocidade e direcção muito particulares, nomeadamente uma intervenção humana. Ou pode ainda o próprio seguidor de trajectórias ele mesmo querer impôr um movimento particular e sobrepor-se ao movimento "sugerido" pelo navegador local.

Portanto, este módulo de administração de movimento terá um tabela de prioridades quanto à origem do movimento a fazer passar para os módulos de real execução do movimento. Na figura da arquitectura (fig. 53), estão representados apenas duas entradas neste módulo: a que vem do seguidor de trajectórias e a do navegador local.

• Geração do movimento

Este módulo é quase exclusivamente passivo. Converte intensidade e direcção do movimento desejado em representações adequadas ao sistema que está subjacente ao robot (neste caso particular em comandos de *Albatros*). A sua existência como bloco independente tem outras razões: poderá ter eventualmente um papel activo e desempenhar acções de filtragem das velocidades a impôr às rodas. Dessa forma poderá evitar situações de mau

funcionamento dos módulos que calculam ou fornecem o movimento a executar. Um exemplo são situações anómalas de geração alternada, a grande taxa, de velocidades elevadas em direcções opostas (situação nunca encontrada na prática, mas possível de impôr ao robot). Questões de integridade do equipamento justificam este cuidado.

• **Navegador local**

O navegador local é um módulo crucial e simples no seu princípio, mas talvez complexo na implementação dos seus sub-blocos. Consiste "simplesmente" em pegar num mapa de percepção, e seguindo uma dada estratégia de movimento, "gerar" uma nova velocidade e direcção para o robot. Por outro lado, a implementação de cada uma das suas estratégias de navegação pode ser elaborada, nomeadamente quanto aos critérios de segurança (distâncias a preservar do ambiente) e fiabilidade dos mapas de percepção usados.

Fig. 58 - O princípio do navegador local.

Este módulo é o garante da componente de autonomia do robot. Prescindindo de qualquer referência de posicionamento absoluto, a navegação que proporciona é do tipo **referenciada no ambiente próximo (local)**. Isto implica que mesmo na ausência de algum dos módulos com funções conceptualmente de mais alto nível, a navegação (o movimento) continua a ser assegurada, pelo menos a curto prazo. Esta propriedade é importante, pois permite interrupções ou inactividade temporária desse tais módulos. Um exemplo é a questão da localização: mesmo sem se saber onde se está em absoluto, continua a ser possível seguir uma dada parede ou progredir mantendo distâncias de segurança aos obstáculos. Outro exemplo da sua importância é o facto de permitir que troços sem grande ambiguidade de trajectória em direcção ao destino final, lhe possam ser confiados à partida sem que os módulos superiores tenham a necessidade de especificar quaisquer velocidades ou pontos de passagem no ambiente.

Este módulo está descrito em maior detalhe nos capítulos precedentes.

• **Construção de mapas de percepção**

A construção de mapas de percepção é o módulo que usa os dados sensoriais (em bruto) e os disponibiliza para o navegador local (ou outros módulos que no futuro os queiram utilizar) sob a forma de mapas de percepção. Nesta aplicação é construído usando redes neuronais apenas, mas pode ser um bloco muito complexo se se usarem muitos tipos de

sensores e a fusão de dados, por exemplo. A construção de mapas de percepção é o assunto por excelência dos capítulos anteriores.

• **Detector de emergências**

O detector de emergências é um bloco que tem associado a si um processo de alta prioridade num sistema de tempo real. A sua função é a de evitar colisões eminentes com os obstáculos. Para isso faz recurso das medidas de ultra-som que estão disponíveis continuamente no sistema. A sua acção é directa e imparável: em caso de emergência detectada, gera de imediato um comando de corte instantâneo dos controladores PID dos motores das rodas. Trata-se portanto de uma acção ao mais baixo nível de controlo.

O critério de avaliação de uma emergência pode ser complexo devido a uma multiplicidade de factores, tais como a velocidade absoluta do robot, a direcção de movimento ou a inactividade ou falha de alguns sensores. Em apêndice (Cf. Apêndice **Um sistema de emergência usando ultra-sons**), são explicadas em maior detalhe as características deste sistema detector de emergências. Todavia, pode-se descrever resumidamente o sistema como sendo um analisador a duas passagens: os sensores definem um anel fechado de medidas, e se uma dada sequência de sensores contíguos nessa anel apresentar medidas inferiores à uma distância critica, então essa sequência define uma emergência potencial. Num segundo passo, essa sequência é verificada segundo alguns critérios: algum dos sensores está desligado? Os sensores da sequência de potencial emergência estão na direcção do movimento? etc. Uma sequência de sensores é um conjunto de sensores adjacentes fisicamente. Testar sequências muito longas (3, 4 ou mais sensores) não garante muita segurança uma vez que os obstáculos podem ser de reduzidas dimensões e podem não cair na zona de influência de alguns dos sensores da sequência. Sequências de um só sensor correm o risco de gerar detecção de falsas emergências. Desta forma, e a prática o mostrou, sequências de 2 sensores são o melhor compromisso entre segurança e detecção de falsas emergências. Porém, se se espera um ambiente muito denso de obstáculos de pequenas dimensões, então, e correndo o risco de falsas detecções, é imperativo considerar cada sensor individualmente e assim evitar com certeza quase absoluta todas a colisões.

É de realçar que os dados sensoriais usados pelo detector de emergências estão na sua forma bruta, isto é, sem qualquer pré-processamento: são usados tal como os sensores se ultra-som os geram.

2.8.2. As interacções do Seguidor de Trajectórias

O Seguidor de Trajectórias, na arquitectura anterior, é um módulo crucial quando se pensa em acções muito concretas de navegação. O seu nome é um pouco restritivo (herança de concepções iniciais) porque o seu papel vai mais além do que forçar o seguimento de trajectórias, até porque isso pode ser complicado e um pouco incompatível com o conceito de Navegação Local aqui desenvolvido, pelo menos quando se pensa em trajectórias rigidamente

definidas. As suas interacções no seio da arquitectura, complexas como já referido, podem contudo resumir-se a dois pontos fundamentais:

- verificar os objectivos da navegação atentando ao desenvolvimento do ambiente e,
- orientar o Navegador Local através da comutação de estratégias de navegação.

O segundo destes dois pontos representa, talvez, a interacção mais simples dado ser uma acção essencialmente unidireccional e o leque de estratégias ser bastante simplificado como já exposto, e também por isso este módulo é cognominado Selector de Estratégias. A situação menos fácil no que respeita à escolha de uma estratégia será o de eliminar ambiguidades da escolha ou de forçar uma estratégia contra-natura (do ponto de vista humano) para evitar comportamentos errados e encaminhar o robot para caminhos igualmente errados (fig. 59).

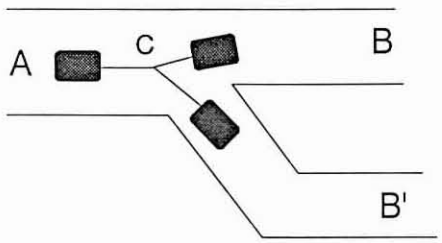


Fig. 59 - "Seguir o espaço livre" pode ser uma estratégia inadequada para ir de A para B. O risco de chegar a B' deve ser evitado forçando uma estratégia diferente (temporária) perto do ponto C.

O real problema das interacções do Seguidor de Trajectórias reside sim na verificação dos objectivos da navegação levando em conta a evolução do ambiente com o movimento do robot. Na implementação prática não houve grandes desenvolvimentos desta componente, mas é possível desde já delinear um conjunto de ideias para desenvolvimentos futuros desta arquitectura de navegação:

- A localização absoluta e suas alternativas.
- Dispensa de execução rígida de uma qualquer trajectória.

A localização absoluta pode ser um problema, sobretudo se não se puder dispor de um conhecimento suficiente do ambiente. Nesses casos poder-se-á recorrer à identificação de configurações particulares desse mesmo ambiente que possam servir de pontos de referência (esquinas, portas ou aberturas, por exemplo). O Seguidor de Trajectórias (Selector de Estratégias) pode ter presente uma lista de pontos de passagem desse tipo, que alguém lhe comunicou no início (ou até modificar dinamicamente) tal como um planeador de trajectória, e usar esses mesmos pontos para uma espécie de localização mais ou menos grosseira, que é quanto baste para efectuar a navegação. Uma localização absoluta e precisa deverá ter outros fins e só será necessária ocasionalmente. É claro que o sistema deverá por certo comportar outros sensores para a percepção dessas referências ambientais, como imagem de vídeo ou um distanciómetro laser, por exemplo. Todavia, se a localização absoluta, a uma taxa suficiente,

for possível, então a cooperação Seguidor de Trajectórias ↔ Navegador Local afigura-se mais simples, como ilustra a seguinte situação:

- Tem-se um dado conhecimento do ambiente (eventualmente incompleto devido à presença de obstáculos desconhecidos ou imprevistos).
- Dispõe-se de um módulo de localização absoluta como mencionado.

Nestas condições, o planeamento de trajectória é possível pelos métodos tradicionais, e o Seguidor de Trajectórias gerará o movimento necessário para a cumprir. A navegação (e o atingir dos seus objectivos) será conseguida com uma execução cíclica dos seguintes eventos:

1. Uma unidade de planeamento calcula a trajectória a executar (ou talvez apenas alguns pontos de passagem) a partir do ponto corrente para o objectivo (ou sub-objectivo intermédio), e o Seguidor de Trajectórias gera o movimento para acompanhar essa trajectória.
2. Na ocorrência de uma situação de emergência (iminência de colisão) ou simplesmente devido à falta de espaço livre na direcção do movimento, o controlo do robot (e a geração do movimento) passa a ser desempenhada pelo Navegador Local, que, com a estratégia adequada, ou combinações (sequências) de estratégias, será capaz de contornar o obstáculo ou de pelo menos desviar-se dele usando o espaço livre que conseguir encontrar. Esta componente foi desenvolvida e testada com sucesso na prática.
3. O retorno do controlo ao Seguidor de Trajectórias far-se-á a qualquer momento quando se verificar uma (ou mais) condições. Exemplos dessa metodologia podem ser os seguintes:
    - Existência de uma deslocação (posição, rotação) **significativa** do robot desde que entrou no modo de Navegação Local. Nesse ponto, refazer localização e voltar ao ponto 1. Caso a deslocação não tivesse sido **suficiente** este ciclo repete-se. Os parâmetros de decisão para esta solução podem ser muito difíceis de afinar (para definir o que é uma deslocação **significativa**) e corre-se o risco de se necessitar de uma grande número de iterações para a recuperação de trajectória ou até mesmo impossível de recuperar a trajectória, caso o robot entre em órbita de um obstáculo. Os dados para concluir sobre a deslocação do robot viriam da **odometria** do sistema.
    - Alternativamente, e admitindo a tal localização absoluta a uma taxa elevada (1Hz seria suficiente para as velocidades esperadas do robot), o critério de retorno do controlo para o Seguidor de Trajectória pode ser o da existência de espaço livre na linha de vista do objectivo de navegação (ou de um sub-objectivo intermédio). Tal como o método

anterior, este processo pode perfeitamente ser iterativo nesta fase de devolução do controlo ao Seguidor de Trajectórias.

Como foi referido anteriormente, e à parte as hipóteses de cooperação lançadas atrás, a concepção de navegação local desenvolvida não é muito propensa à execução rígida de uma trajectória. Esse tipo de tarefa obriga necessariamente, em robótica móvel, à existência de um processo de localização absoluta e precisa, e quase contínua (para saber se se está ou não na trajectória!). Assim, o Selector de Estratégias não é tanto um seguidor de trajectórias rígidas (se bem que na arquitectura proposta pode sempre fazê-lo de sua iniciativa e apoiada pelo módulo de localização), mas sim um determinador de pontos aproximados de passagem — o Navegador Local fará a interpolação desses pontos e gerará uma trajectória, não óptima nem talvez exactamente previsível, mas será certamente bastante segura.

Em resumo, o que se pode esperar do Seguidor de Trajectórias e Selector de Estratégias é, em termos um pouco idealistas, cumprir uma ordem do tipo: "**ir pelo corredor além, virar na 3ª porta à direita e seguir em frente até à sala ao fundo!**". É claro que os obstáculos (inesperados) que se lhe deparem não devem alterar esta configuração de ambiente ao ponto de, por exemplo, não se poder seguir em frente após a tal 3ª porta. Supõe-se também a existência do tal sistema complementar (à base de outros sensores, em princípio) que saiba identificar uma porta quando passa por ela.

Para finalizar, e delineando já linhas de evolução para o desenvolvimento do módulo do seguidor de trajectórias, propõe-se-lhe uma subdivisão interna como ilustrado na figura 60.

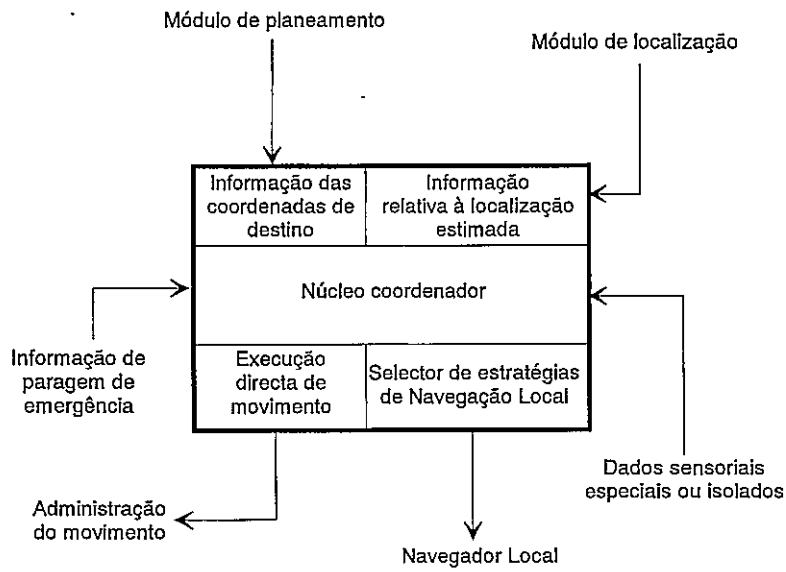


Fig. 60 - Proposta de divisão interna do Seguidor de Trajectórias e Selector de Estratégias.

Com uma substrutura deste tipo, o seguidor de trajectórias estará em condições de efectuar seja a selecção adequada de estratégias para levar o robot para uma dada região (sem grande precisão mas também sem o mínimo esforço para gerar e calcular o movimento a impor), seja a execução rigorosa de comandos concretos de movimento. Esta última

possibilidade permitirá a execução de trajectórias eventualmente calculadas por um módulo de planeamento, para as quais recorrem de igual modo os dados relativos à localização. Permite ainda a execução de manobras especiais como por exemplo o estacionamento (*docking*) mais ou menos preciso. Para isso poderá recorrer a dados sensoriais isolados como sejam dados de alguns sensores de ultra-som para, por exemplo, estacionar paralelamente a uma parede a uma dada distância desta.

Não é contudo de excluir a possibilidade da inserção na arquitectura proposta de módulos de navegação dedicados a manobras particulares e portanto altamente especializados na sua execução, como o caso do *docking*. Todavia, a introdução de novos módulos deve ser bem ponderada para continuar a permitir a contínua evolução e adaptação da arquitectura mantendo as compatibilidades entre os módulos existentes.

Parte 3

Resultados experimentais e conclusões

3.1. Introdução

Esta parte começa por descrever os resultados obtidos por cada uma das técnicas para os diversos problemas surgidos e descritos nas partes anteriores. Esses problemas vão desde a maximização da taxa útil de disparo dos sensores de ultra-som, passando entre outros pela convergência das redes neuronais até ao ajuste dos diversos parâmetros das estratégias de navegação.

As conclusões dos resultados experimentais fazem um balanço entre os problemas expostos no início e os problemas efectivamente resolvidos com as técnicas e os métodos propostos. São ainda abordados os novos problemas deixados em aberto e, na medida do explorado, linhas de acção para os ultrapassar.

3.2. Melhorias nos dados de ultra-som

Os dados de ultra-som são, sem qualquer dúvida, uma das componentes mais importantes deste trabalho; são esses dados sensoriais que permitem a percepção do espaço em torno do veículo, e é também com base neles que se desenvolve um método para a detecção de eminência de colisão (ver Apêndice - **Um sistema de emergência usando ultra-sons**).

Além da problemática própria que caracteriza os sensores de ultra-som ao nível da interpretação das medições (ver **Parte 1**), permanecem acima de tudo as limitações intrínsecas ao seu princípio físico e que se podem resumir numa só: a reduzida velocidade de propagação do som no ar! Este facto acarreta o grave problema dos "longos" tempos de medição. Se isso pode não parecer problemático quando se trata de um sistema com um só sensor, a situação agrava-se quando o número de sensores a utilizar (simultaneamente) cresce: ou se aceita uma baixa taxa de medição sem interferências entre sensores, ou se corre o risco certo de interferência subindo a taxa de medição!

É assim vital garantir a maior qualidade possível dos dados sensoriais, sobretudo quando esses dados pretendem ser representativos ou indicadores de determinadas situações, como ocorre com os dados adquiridos nas sessões de aquisição, para mais tarde serem usados no treino de redes neuronais. Em resumo, pretende-se a máxima taxa de disparo dos sensores e minimizar simultaneamente os problemas de interferência; para isso exploraram-se dois conceitos: o agrupamento de sensores e as estratégias de disparo múltiplo.

3.2.1. Agrupamento de sensores e estratégias de disparo múltiplo.

A interferência entre sensores (*crosstalk*) ocorre quando um sensor detecta o eco de um feixe que não foi emitido por esse próprio sensor. Essa situação ocorre necessariamente nos períodos de "escuta" dos sensores, isto é, no intervalo de tempo que decorre entre a emissão de um feixe e a detecção de um sinal supostamente de retorno (na mesma frequência, geralmente). Os riscos de interferência são muito elevados quando dois ou mais sensores são disparados simultaneamente.

A interferência pode ser directa ou indirecta. No primeiro caso, um sensor recebe o feixe emitido por outro sensor directamente ou, possivelmente, pela propagação através das estruturas mecânicas de suporte. A interferência indirecta surge quando um sensor detecta o eco destinado a outro sensor, portanto detecta um feixe reflectido que foi emitido por outro sensor.

Para a minimização da interferência entre sensores disparados simultaneamente (ou com intervalos de tempo o mais curto possível) é de grande importância ter em conta a orientação relativa desses sensores. São assim de procurar as configurações onde esses sensores simultâneos "apontem" em direcções opostas, ou pelo menos em direcções o mais divergentes possível (fig. 61).

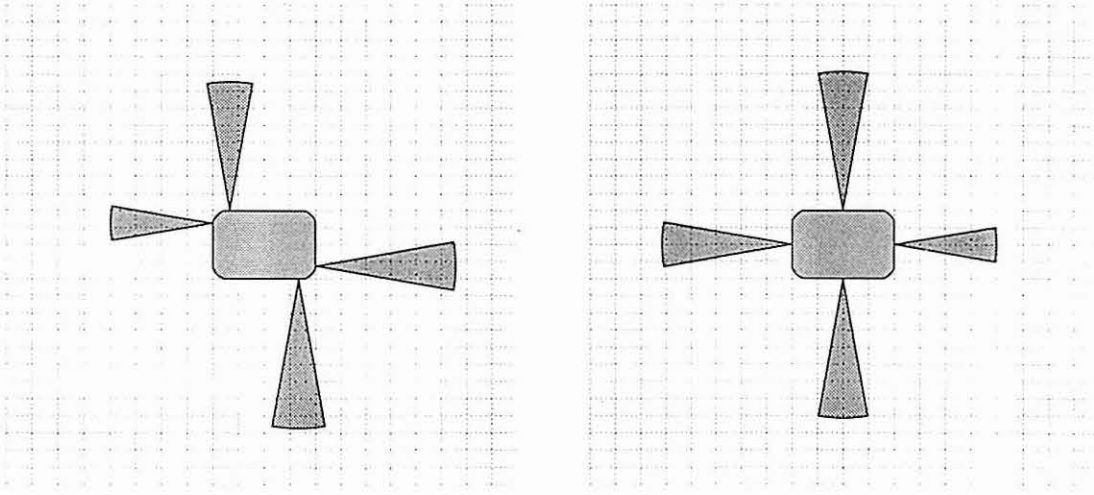


Fig. 61 - Ilustração de dois nodos de sensores de ultra-som no *Robuter*.

Se se assumir que as "direcções" de sensores simultâneos devem divergir de pelo menos 90°, tem-se a possibilidade de ter, no máximo, 4 sensores a serem "disparados" ao mesmo tempo. Para um sistema com 24 sensores e dotado geometricamente de dois planos de simetria (como é no *Robuter*) este processo resulta em 6 grupos de 4 sensores ortogonais nas suas direcções, isto é, é possível disparar os 24 sensores por 6 vezes com um risco diminuído de interferência. Estes grupos são designados por **nodos** e estão previstos no próprio sistema fornecido comercialmente. Contudo, estas associações não são suficientes para obter uma taxa elevada de medição por uma razão constatada experimentalmente: os 4 sensores disparados simultaneamente provocam ainda interferência entre si. O disparo simultâneo é o grande

óbice! Aparece assim comprometida a expectativa de bons tempos de medição (se disparamos os 6 nodos com um intervalo de 50 ms entre cada um ter-se-á uma frequência de medição global superior a 3 Hz [1000/(6×50)] — os 50 ms permitem medição de distâncias relativamente longas [até ≈ 8.5 m] o que reduz significativamente o risco de interferência entre sensores de nodos diferentes.)

Dada esta desvantagem relativamente ao disparo simultâneo de sensores, mesmo quando "divergentes" entre si, tornou-se necessário conceber métodos de disparo onde um número mínimo de sensores fosse disparado em simultâneo. As técnicas usadas resultam em combinações múltiplas dos seguintes factores:

- Intervalo de tempo entre nodos (0,1,2,3,... *ticks* (unidades) do relógio de tempo real — um *tick* de relógio de tempo real neste sistema era de 7.5 ms)
- Ordem de disparo dos 6 nodos.
- Subdivisão de um nodo em subnodos e a sua ordem de disparo.
- Intervalo de tempo entre os subnodos (0,1,2,3,... *ticks* do relógio de tempo real)

As possibilidades de sequências e subnodos testados estão ilustrados na tabela 2.

Sequências de nodos
Directa - 123456
Inversa - 654321
Pares primeiro - 246135
Ímpares primeiro - 135246
Aleatória

a)

Sequências dos 4 sensores dentro de cada nodo (subnodos)
Simultânea: 1+2+3+4
Sequência directa: 1 2 3 4
Sequência inversa: 4 3 2 1
Sequência alternada: 1+3, 2+4

b)

Tabela 2 - Sequências possíveis de disparo dos nodos e dos sensores em cada nodo (subnodos). Os intervalos de tempo entre sequências são configuráveis.

Das possibilidades listadas na tabela 2a, é de salientar que os resultados com diferentes sequências de nodos não diferiam muito para um dado intervalo de tempos entre nodos; ou o intervalo de tempo era suficiente (pelo menos 5 *ticks* de relógio de tempo real funcionava bem na quase totalidade das situações) ou havia risco de interferência entre sensores de nodos disparados consecutivamente. A componente mais significativa na estratégia de disparo dos sensores, como já era de esperar, reside na sequência de subnodos (tabela 2b). Disparar um sensor de cada vez (4 subnodos, portanto) foi o que providenciou melhores resultados (sequência directa 1 2 3 4 ou inversa 4 3 2 1), mas a situação mais favorável exigia cerca de 450 ms (medidos experimentalmente) para completar uma ronda completa pelos 24 sensores.

Resultados de reduzida interferência foram também conseguidos com subnodos de 2 sensores (sequência alternada) e os tempos globais de medição caíram abaixo dos 300 ms. Tem-se assim que é possível, na grande maioria das situações experimentadas (centenas de horas de navegação e medição ultra-sónica), efectuar medições com 24 sensores de ultra-som

a uma taxa de pelo menos 3 Hz com um risco baixíssimo de interferência entre sensores — visualmente† era quase indetectável a existência de interferência: grosso modo notava-se uma interferência em cerca de 5 minutos (1000 disparos), e especulando que a percepção e atenção humana perdesse 90% das interferências realmente ocorridas, o nível de interferência estaria mesmo assim em torno de 1% dos disparos. Note-se que os resultados tradicionais com sistemas semelhantes poucas vezes conseguem tempos de medição inferiores a 500 ou 600 ms [Borenstein92], se bem que trabalhos mais recentes de J. Borenstein e Y. Koren [Borenstein95] apresentem melhorias muito significativas recorrendo a métodos de rejeição de leituras erróneas dos sensores. A interferência seria significativamente reduzida se se usassem sensores em frequências ultra-sónicas diferentes entre os diversos sensores, isso permitiria taxas os disparos simultâneos desses sensores.

Como exemplo, tem-se que uma das combinações mais bem sucedidas foi: uma **sequência directa de nodos** com atraso de 5 *ticks* de relógio (37.5 ms) entre disparo de nodos sucessivos, e **sequência alternada de subnodos** disparados com o **menor intervalo de tempo** que o sistema permitisse (na prática variava entre 1 e 2 *ticks* de relógio), o que perfaz o tempo total de aproximadamente 6 × (37.5 + 10) = 285 ms para um ciclo completo de disparo dos sensores.

As raras situações onde ainda se verificaram problemas de interferência estava associada a configurações particulares do ambiente. O caso detectado ocorria sempre no mesmo local da sala, e era uma situação onde o robot estava muito próximo de um canto dessa sala e com uma dada orientação (fig. 62). Essa situação compreende-se pelas curtas distâncias a medir e pela orientação perfeitamente propícia à reflexão e ao desvio directo do feixe de um sensor para outro.

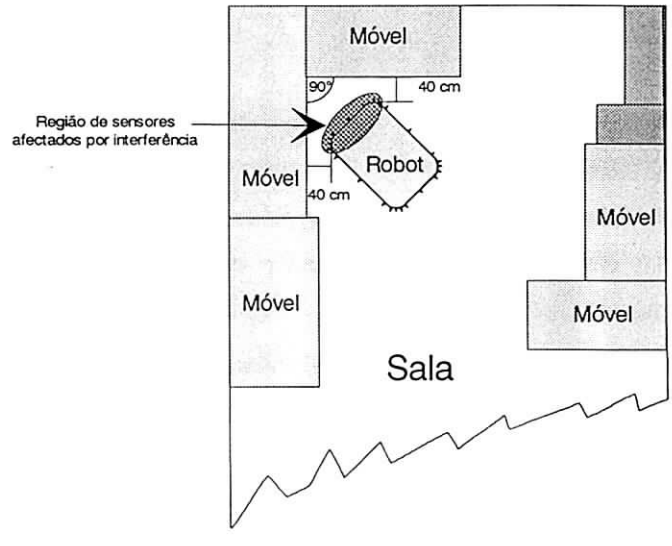


Fig. 62 - Uma situação de interferência de sensores verificada experimentalmente e devido certamente à orientação do robot numa geometria particular do ambiente.

† O controlo visual era feito por meio de uma ferramenta desenvolvida para a representação de medições ultra-sónicas. Ver em apêndice uma referência a essa ferramenta.

Na prática, melhorar a taxa de disparo podia consistir também em disparar apenas os sensores necessários à navegação. Os mapas de percepção que se usaram necessitam apenas de 15 sensores, donde não há necessidade imediata de disparar os outros 9 sensores. A excepção a este caso é a informação para a detecção de emergências (colisões); na verdade há sensores disparados mas não usados para o cálculo destes mapas de percepção em particular, mas que podem salvaguardar uma colisão da traseira do veículo durante uma rotação por exemplo. O disparo de sensores na prática era regulado por uma tabela instantânea dos sensores que era modificada dinamicamente de acordo com a direcção do movimento (velocidade relativa das rodas). É claro que se se usar uma mapa de percepção completo em redor do veículo então os sensores terão que ser todos usados.

3.3. Treino das redes neuronais

Como foi adiantado na **Parte 2**, foram experimentadas duas formas diferentes de usar redes neuronais para resolver o problema da geração do mapa de percepção a partir dos dados sensoriais de ultra-som. O primeiro método consistia em usar a rede como um aproximador de funções contínuas (a função de ocupação de células numa GGG usando os valores sensoriais como variável independente), e o segundo método propunha uma reformulação do problema em termos de uma classificação de vectores contínuos (os dados sensoriais) num número limitado de padrões de ocupação (classes). A legitimidade dos dois métodos foi já explicada, e a sua aplicabilidade é aqui distinguida através dos exemplos experimentados.

3.3.1. As dimensões das redes e o algoritmo de treino

A seguir à escolha da arquitectura de uma rede, o problema que se põe é, nos casos onde isso se aplica, as dimensões da ou das camadas escondidas. Não existem ainda leis precisas para a determinação do número das unidades escondidas. Excluindo a perspectiva matemática do teorema de Kolmogorov, restam umas certas regras semi-empíricas baseadas na dimensão do conjunto de treino: por vezes invoca-se que o número de conexões (pesos) em toda a rede não deve ser inferior ao número de elementos do conjunto de treino, ou ainda que o número de unidades escondidas (uma só camada) não deve ser menor que uma ordem de grandeza que o número de elementos do conjunto de treino, mas há muitas redes a funcionar que se desviam destes requisitos. Vários estudos têm sido feitos, como em [Baum88], [Mirchandani89] ou [Huang91] entre outros, mas, até ao momento, a dimensão adequada de uma rede é obtida essencialmente por via empírica, ou seja a experimentação. Redes demasiado pequenas são aquelas que não "aprendem" o conjunto de treino, e redes demasiado grandes não adquirem propriedades de generalização (aprendem bem a reconhecer apenas o conjunto de treino).

O algoritmo que se usa em redes multi-camada como as usadas (também actualmente conhecidas por Perceptrões multi-camada) é o algoritmo dado a conhecer por Rumelhart *et al.* num famoso artigo da revista *Nature* em 1986 [Rumelhart86]: trata-se do *Back-progagation of errors* (retropropagação do erro) ou simplesmente *back-propagation* (retropropagação). Este

algoritmo foi depois modificado e complementado várias vezes; entre elas realce-se a introdução de um termo de inércia de aprendizagem que permite acelerar a convergência do algoritmo base: trata-se do **"retropropagação com termo do momento"** [Vogl88]. Deste modo, os parâmetros mais importantes para o algoritmo de retropropagação disponível no simulador usado são: o coeficiente de aprendizagem a que pode ser dinâmico (o simulador usado não o permitia) com valores inferiores a 0.1 recomendados para o início de um treino, e o termo do momento i, onde os valores comuns variam entre 0.1 e 0.8, mas sem um regra rígida. Nos gráficos seguintes os valores usados para esses parâmetros, em cada caso, aparecem junto do próprio gráfico.

3.3.2. Redes de saídas contínuas

O critério de convergência inicial foi simplesmente o do erro médio da saída: sempre que se atingisse uma condição em que a média do erro dos padrões apresentados fosse inferior a um dado limite, dizia-se que a rede tinha convergido. Os dados de treino assentaram nos dados reais de ultra-som normalizados ao intervalo [0, +1]. Assim os padrões de treino são pares de vectores **dados/valor ocupação** com a seguinte formatação:

dados $S = (s_0, ..., s_{N-1})$ e valor ocupação $O = (o_0,...,o_{M-1})$

com N=24 e M=60 para uma rede global e N=3 e M=8 ou M=6 para as redes parciais menores.

$s_j \in [0,+1[$ e $o_k \in \{0,+1\}$, *que são os valores de ocupação de cada célula.*

3.3.2.1. Uma rede neuronal global

A primeira rede experimentada, como foi já referido, pretendia fazer uma correspondência entre as 24 medidas sensoriais e as 60 células do mapa de percepção. A dimensão da camada escondida que melhor satisfazia as condições foi de 140 neurões. Valores menores e maiores foram também testados.

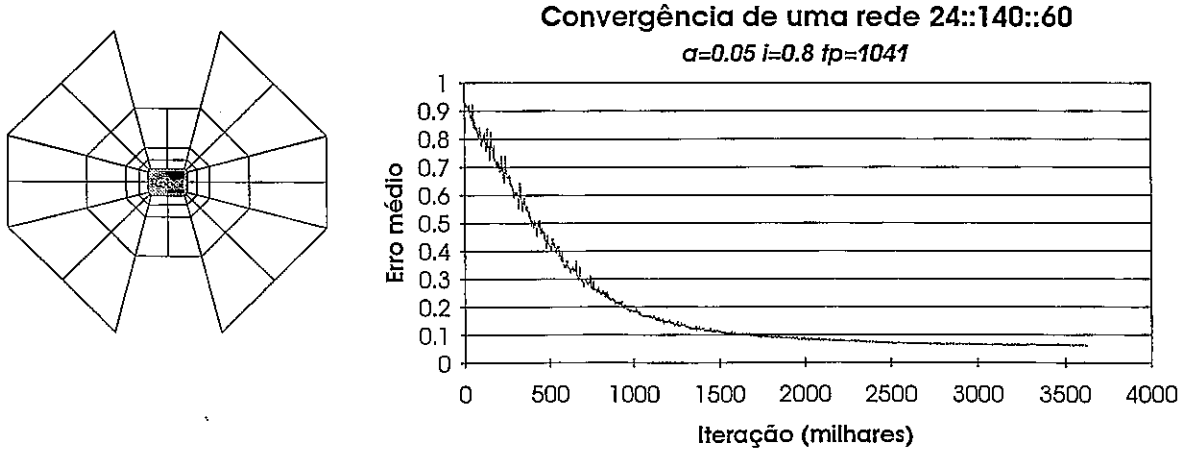


Fig. 63 - Curva da convergência (erro médio) de uma rede 24::140::60 de ligações totais e inteiramente contínua usando o algoritmo de retropropagação com momento.

A rede neuronal global (24::140::60) convergiu e um exemplo do seu comportamento pode ser apreciado na figura 64. A rede treinada é de seguida aplicada sobre dados reais, completamente novos vindos do robot; no exemplo da figura 64 nota-se que a rede constrói um mapa onde algumas reflexões especulares são contornadas e a representação do espaço livre obvia as falhas sensoriais. Nesta altura os resultados surgem bastante encorajadores.

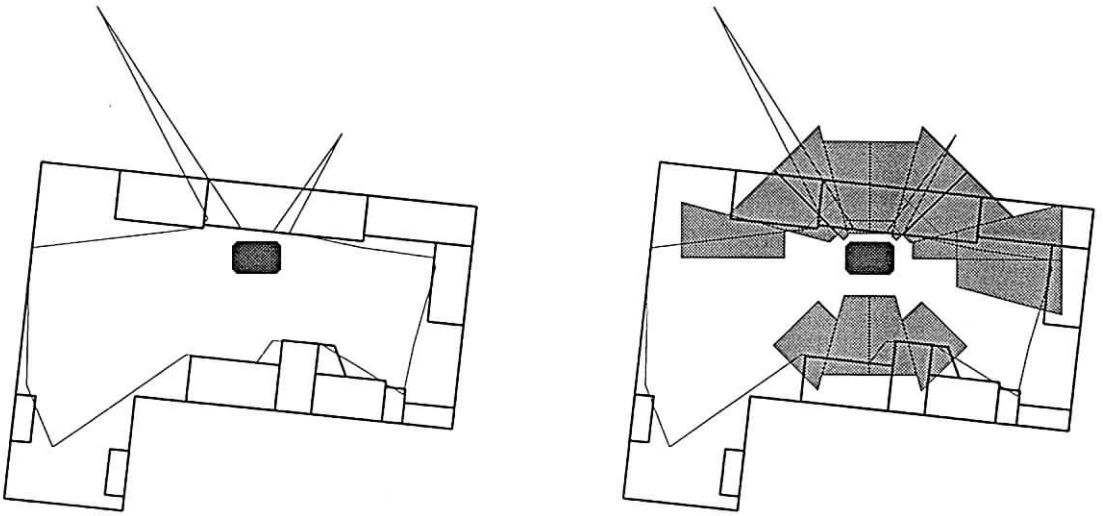


Fig. 64 - Resultados de construção de mapa de percepção com uma rede neuronal global onde se nota a eliminação de reflexões especulares. a) Sobreposição das medidas sensoriais em bruto com a modelo da sala. d) Sobreposição das medidas, do mapa de percepção gerado pela rede, e do modelo da sala.

Os resultados obtidos com uma rede global eram interessantes como se verificou pelos testes levados a cabo. Para confirmar se a rede apresentava boas capacidades de generalização

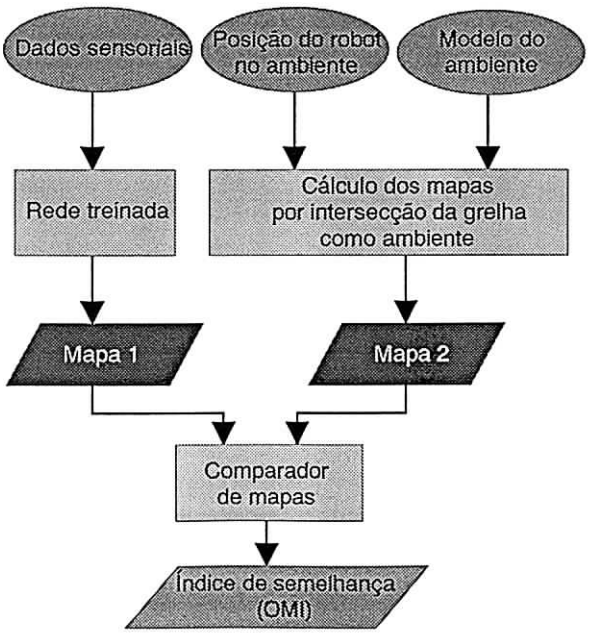


Fig. 65 - Cálculo de um índice de semelhança (OMI) entre mapas gerados por uma rede neuronal global e os mapas que a rede era suposta gerar.

fez-se uma experiência de navegação real operada manualmente, onde se supunha o robot sempre localizado e se comparavam os resultados dos mapas gerados pela rede com os mapas que a rede seria suposta gerar. Esses supostos mapas eram possíveis de ser calculados uma vez que se tinha o modelo do ambiente e a posição nesse ambiente (esta última sempre sob vigilância manual dada a propensão do sistema de odometria para a acumulação de erros, como já descrito anteriormente). A experiência e os resultados vêm sumariamente descritos em [Santos93] e [Santos94], mas o esquema montado pode ser rapidamente ilustrado como indicado na figura 65.

O cálculo do **índice de semelhança de ocupação** (*Occupancy Matching Index* - ***OMI***) era baseado na contabilidade de células correctas do mapa gerado pela rede em relação ao mapa calculado geometricamente:

$$OMI = \frac{Células\ correctas}{Número\ total\ de\ células\ \ da\ grelha}$$

Deste modo, o OMI era um indicador com alguma validade em termos médios, como se pode ver na figura 66. As camadas de células 2 e 3 do mapa são as que cobrem as distâncias intermédias em torno do robot.

Na figura 66 pode-se ver que os mapas tinham em média uma contagem de células correctas em torno dos 92-93%. Todavia, isto é insuficiente; havia muitos mapas correctos a 100%, mas havia alguns correctos a apenas 60 ou 70%, valores demasiado baixos para garantir um percepção clara do ambiente em todas as circunstâncias. Mais uma vez se constataram as limitações indicadas anteriormente: os mapas errados ocorriam sobretudo em situações onde o robot se aproximava dos obstáculos.

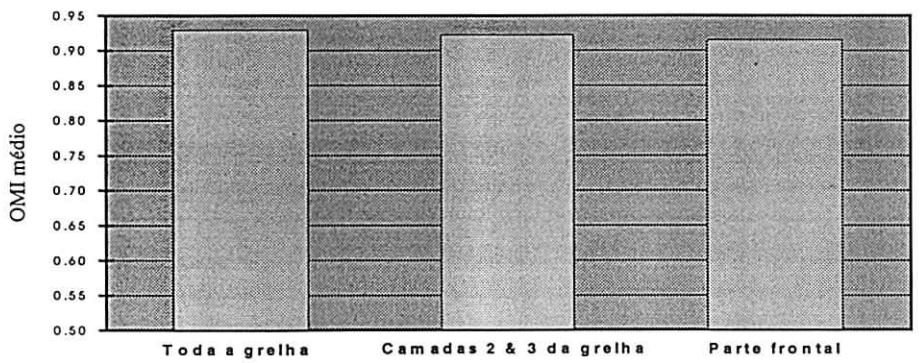


Fig. 66 - Valores médios do OMI ao longo de sessões de navegação manual. Estão ilustrados os cálculos para o mapa todo, para as camadas 2 e 3 apenas, e para a parte frontal do mapa apenas.

Neste ponto havia ainda uma solução a tentar: arranjar um conjunto de treino que cobrisse mais situações de proximidade com os obstáculos, e depois treinar a rede e habilitá-la a lidar com essas situações. Para o efeito foram levadas a cabo novas sessões de aquisição onde se procurou (com grande preocupação de segurança) navegar junto dos obstáculos. Os conjuntos de treino resultantes foram usados em redes semelhantes às descritas, mas os

resultados foram apenas ligeiramente melhores: a rede convergiu de forma idêntica, o OMI médio era ligeiramente mais alto, mas havia ainda situações que pareciam não estar contempladas no conjunto de treino: a rede era ainda incapaz de gerar bons mapas em situações de vizinhança de obstáculos.

3.3.2.2. Outras redes neuronais contínuas (parciais)

Dadas as dificuldades em definir uma rede global com boas propriedades de generalização, e como foi já adiantado na **Parte 2** (proposta de navegação), a questão transformou-se então em subdividir o problema, para diminuir assim o número de possibilidades que uma rede devesse aprender. Em vez de se possuir uma rede única, tentaram-se redes que cobriam regiões menores do espaço em torno do robot. Seguiram-se como é obvio os divisões intermédias da grelha, e usaram-se os dados sensoriais que cobriam essencialmente essas zonas e que tinham sido adquiridos nas sessões de aquisição anteriores. Mostram-se adiante os resultados dessas experiências: as convergências ilustradas mostram por vezes situações com longos tempos de treino, isto é, com milhões de iterações, ou seja milhares de épocas, uma vez que os conjuntos de treino possuíam em geral 1000 a 3000 pares de padrões de treino. As exigências de convergência tinham de ser grandes (seguras) uma vez que se envolverão questões de segurança do robot, daí os longos períodos de aprendizagem. Muitas redes foram experimentadas em situações ainda longe da convergência exigida (com, por exemplo, 80% dos padrões aprendidos), e, de facto, as redes apresentavam a imaturidade esperada; geravam mapas de percepção incorrectos, mas não na proporção de 20% como se poderia esperar se a rede só tinha aprendido bem os tais 80% dos padrões: em algumas redes era mais do que isso e noutras menos, dependia de igual forma das condições de teste (posição na sala), porque também não é previsível quais são os padrões que a rede aprende primeiro!

A certa altura, foi também tentado um método complementar de monitorização da convergência para evitar o risco de sobreaprendizagem da redes: após um determinado número de épocas de treino, a rede era testada sobre um conjunto de teste separado previamente do conjunto de padrões de treino. Quando o número de padrões falhados (não reconhecidos) desse conjunto de teste passasse por um mínimo e apresentasse tendência para subir, a rede estaria a entrar em sobreaprendizagem! Dos vários testes feitos não resultou evidência de sobreaprendizagem, talvez porque os dados reservados para o teste eram muito similares aos dados do treino.

3.3.2.2.1. Frente e lados - 5 cones

Uma primeira tentativa para simplificar a rede consistiu em tentar uma que gerasse um mapa mais reduzido, fundamentalmente a parte mais interessante para a navegação: a frente e os lados.

Como se pode ver pela figura 67, a rede para um mapa consistindo apenas na parte frontal e partes laterais do mapa global convergiu (se bem que exigiu ainda uma grande

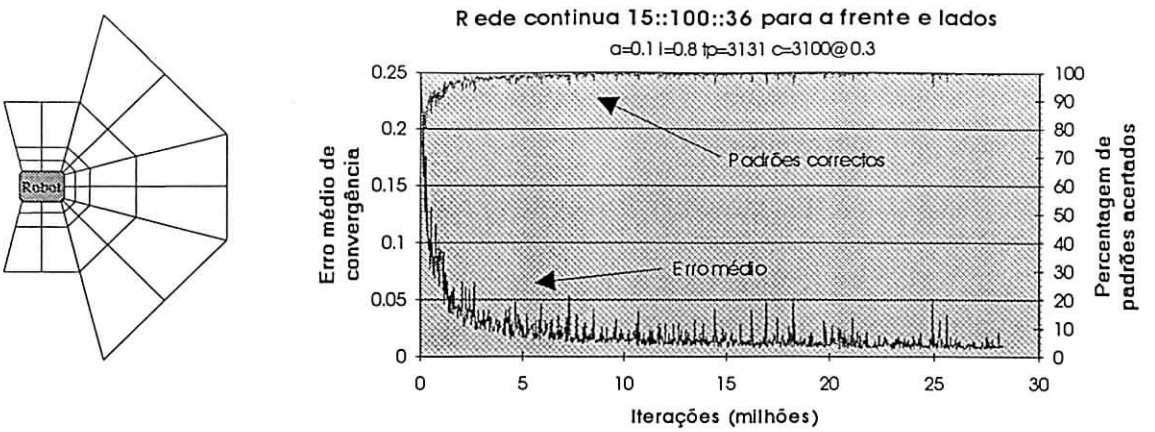


Fig. 67 - Convergência de uma rede (à direita) para um mapa reduzido (à esquerda).

camada escondida e com um longo tempo de treino). Porém, de novo se verificou na prática uma incapacidade de generalizar: o mapa é ainda muito grande oferecendo muitas possibilidades e exigindo um conjunto de treino muito difícil de obter experimentalmente (para ser bem representativo). A única forma de fazer convergir uma rede para um mapa destas dimensões foi usar uma camada escondida grande (100 neurões) o que em si pode reduzir muito a capacidade de generalização levando muitas vezes ao que se refere como um rede que "memoriza" um conjunto de treino e reage mal a novos dados. É também possível que possa ter havido alguma sobreaprendizagem do conjunto de treino pelo elevado número de épocas, mas isso é improvável uma vez que a rede generalizava desde que as situações de navegação não envolvessem muita proximidade aos obstáculos.

3.3.2.2.2. Parte frontal - 9 sensores para 24 células

Resultados semelhantes ao caso anterior ocorreram para uma rede que deveria gerar um mapa ainda mais reduzido que nesse caso: agora, os 9 sensores da zona frontal e as respectivas zonas cobertas num total de 24 células formaram o conjunto de treino. O mapa é mais

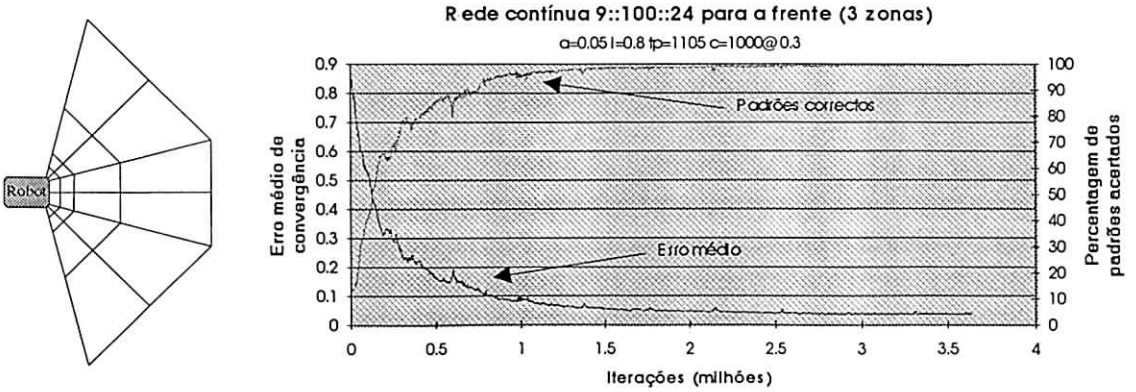


Fig. 68 - Convergência de uma rede para a geração de uma mapa reduzido consistindo apenas na parte frontal do mapa global.

pequeno do que nos casos anteriores e a rede também, mas os problemas permaneciam (em ligeiramente menor escala) e a explicação encontrada é ainda a mesma (dificuldade em obter um conjunto de treino representativo), e de novo excluindo a possibilidade da sobreaprendizagem pelas mesmas razões enunciadas anteriormente.

3.3.2.2.3. Cones (trapezoides) individuais

Prosseguindo a lógica de ir reduzindo o tamanho dos mapas para se chegar a uma situação onde se obtivessem redes capazes de generalizar, chega-se ao caso limite, nesta abordagem: pretende-se agora uma rede que transforme 3 leituras sensoriais em valores de ocupação da região coberta por esses 3 sensores (os trapezóides ou cones, como também foram já designados).

Aqui o problema encontrado foi outro: foi impossível fazer convergir qualquer rede com as mais variadas dimensões da camada escondida. Há 3 situações diferentes entre si correspondendo a cada tipo diferente de trapezóide no mapa global, como foi já referido anteriormente.

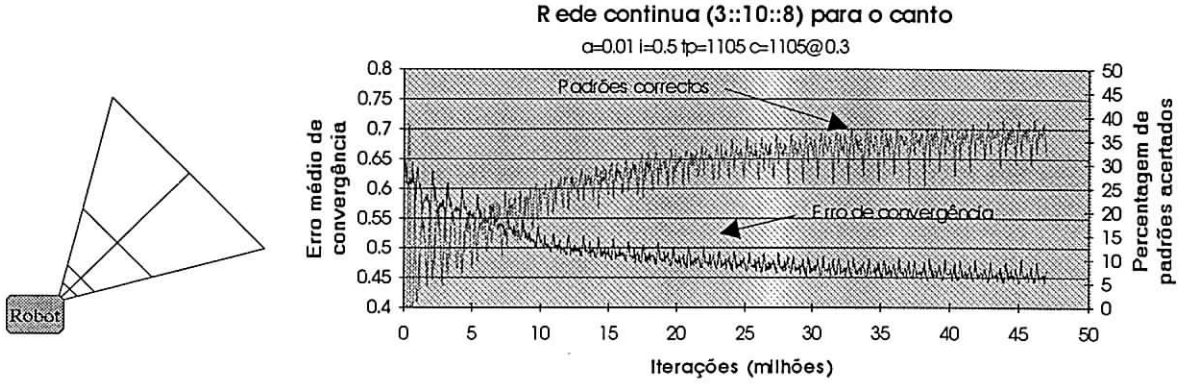


Fig. 69 - Exemplo da (não) convergência de uma rede parcial 3::25::8 para uma região de canto.

Na verdade, foi impossível determinar uma rede neuronal que convergisse com um critério minimamente razoável: conseguiram-se redes que aprenderam a reconhecer cerca de 75-80 % (no exemplo da figura 69 menos de 60%) dos padrões apresentados (alguns milhares em certos casos) com um erro de 0.3 por padrão (no intervalo [0,+1]); o erro de um padrão está definido na **Parte 2**. Porém, estes valores de convergência são visivelmente insuficientes, nomeadamente porque os padrões não reconhecidos correspondiam a situações de pouco espaço livre (ou seja, os casos críticos para um modelo de representação do espaço em torno do robot!). Note-se de igual modo que o erro médio de convergência manteve-se também bastante alto.

É pertinente pôr a seguinte questão: porque é uma rede de grandes dimensões (24::140::60), quando comparada com estas (3::*x*::8, com *x* entre 0 e 50, como tentado), converge e estas menores não convergem? A pertinência é ainda maior quando pensarmos que os dados para os respectivos conjuntos de treino derivam do mesmo conjunto de dados

adquiridos experimentalmente. A resposta empírica foi a seguinte: os conjuntos de treino para as redes menores continham inconsistências estatísticas!. Isto quer dizer que padrões de entrada similares (isto é, é pequena a distância entre os vectores que os representam) tinham como correspondência valores de ocupação (os tais determinados a partir da real posição do robot) muito díspares entre si† .

Esta disparidade de padrões (ocupação de células) deve-se essencialmente a uma abundância de reflexões especulares, que se poderia ousar contabilizar nos 20-25 % dos dados (uma vez que era essa a percentagem de padrões não ensináveis à rede nas melhores condições conseguidas). As reflexões especulares originam medidas maiores de distâncias do que as verificadas da verdadeira posição do robot: o problema fulcral das reflexões especulares é que, em condições semelhantes, nem sempre ocorrem — dependem de pequenas variações da orientação do robot e até de um certo comportamento estatístico nas medições efectuadas por um sensor. No caso da rede global (ou até redes para mapas maiores que estes simples cones) era muito mais difícil de encontrar padrões contraditórios dada a dimensão do padrão de entrada (24 valores contínuos para a rede global): haveria sim padrões incorrectos pelas reflexões especulares, mas os padrões contraditórios seriam em muito menor número porque o universo de possibilidades era muito maior, e a rede conseguiu estabelecer uma relação entre o conjunto global (todos os sensores e todas as células). Por outro lado, porque o universo de possibilidades era tão grande, a rede não conseguiu obter as propriedades de generalização necessárias.

Igualmente logradas foram as tentativas de fazer convergir uma rede para submapas bastante pequenos (um trapezóide) mas usando 5 dados sensoriais em vez de 3. Ou seja, aos 3 sensores correspondentes à zona em causa foram acrescentados os 2 sensores adjacentes — um de cada lado. Como já mencionado anteriormente, pretendia essa facto tentar alargar as capacidades da rede e levar em conta uma qualquer espécie de continuidade geométrica do ambiente e assim suplantar eventuais leituras faltosas de sensores. O princípio não pôde ser verificado talvez pelos problemas mencionados atrás acerca dos dados contraditórios, mas é um problema que pode ficar em aberto para futuros desenvolvimentos de construção de mapas de percepção usando redes neuronais.

3.3.2.2.4. Dados sintetizados

Uma vez que os dados sensoriais apresentavam inconsistências estatísticas, para verificar se o princípio do uso das redes neuronais estava ou não comprometido, experimentou-se sintetizar dados de ultra-som sem as tais inconsistências.

O princípio da síntese de dados pretendeu ser simples: tratava-se de gerar os dados sensoriais e as ocupações celulares que lhe corresponderiam. Neste processo não há um tratamento específico de reflexões especulares, mas isso é feito de forma indirecta pela análise

†Agradecimentos em particular ao Dr. J. Larrisse do CCR-Ispra por ter fornecido essas conclusões.

das próprias medidas sensoriais sintetizadas usando umas regras semelhantes às usadas na geração de mapas de percepção por um dos métodos alternativos propostos anteriormente (ver **Parte 2**).

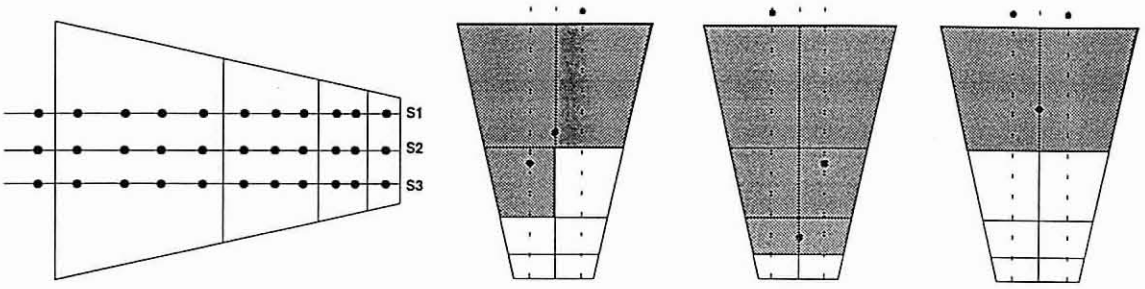


Fig. 70 - Ilustração da síntese de dados sensoriais. À esquerda os pontos de amostragem das distâncias, e à direita 3 exemplos concretos de ocupação das células.

Assim, a síntese assenta na definição de um determinado conjunto de medidas sensoriais dos 3 sensores na área de influência de um cone (trapezóide). Procurou-se que essas medidas sensoriais caíssem longe das fronteiras das células para evitar situações de ambiguidade ou imprecisão de ocupação dessas mesmas células (fig. 70). No exemplo da figura teríamos assim 12×12×12=1728 combinações de medidas sensoriais. As medidas do sensor central pesam em todas as células da região.

Uma outra vantagem dos dados sensoriais sintetizados é cobrirem a gama que se desejar. Não se está mais limitado aos problemas práticos da aquisição. É portanto mais fácil garantir uma distribuição mais equitativa e mais representativa dos dados. Também estes factores foram levados em conta na síntese de dados para a aprendizagem das redes.

Treinaram-se então as mesmas redes de pequenas dimensões (um só trapezóide) com os novos dados sintetizados. Os resultados foram excelentes no sentido em que as redes convergiram bastante bem como se pode ver na figura 71. Nessa figura estão dois exemplos das várias experiências. As diferenças aparentes entre os dois casos resultam de escalas diferentes num lado e no outro. Pode-se afirmar contudo que a rede 3::25::8 progrediu mais rapidamente nas primeiras iterações do que a rede 3::10::6. A primeira era para regiões dos

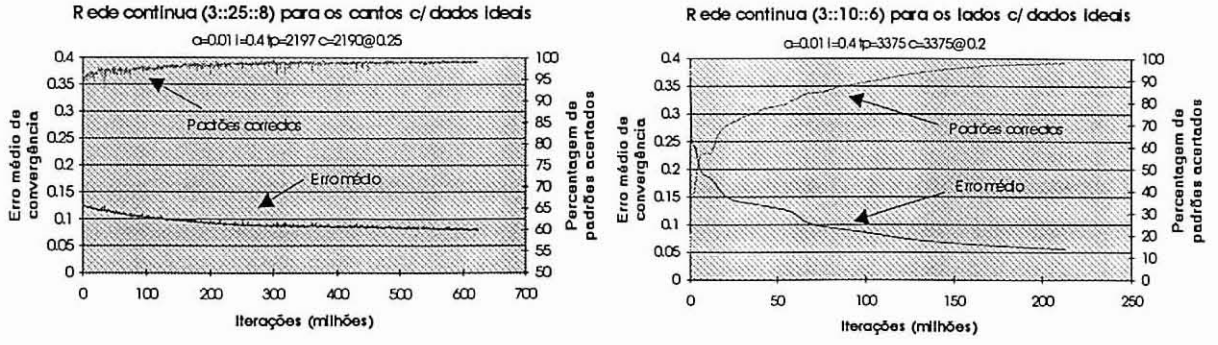


Fig. 71 - Exemplos da convergência de redes contínuas usando dados sintetizados.

cantos (8 células) e a outra para as regiões laterais (6 células). Essa diferença de comportamento inicial tem a ver sobretudo com as dimensões das redes: no caso das regiões laterais, a rede só tem 10 neurões na camada escondida, o que obrigou o algoritmo a mais iterações dados os recursos proporcionalmente mais reduzidos do que no outro caso (25 neurões na camada escondida).

Estava assim provado que parte dos problemas encontrados com os dados reais eram devidos às contradições existentes nesses mesmos dados.

3.3.2.3. Saídas sem significado físico

Uma questão associada a esta forma de usar as redes como interpoladores contínuos de ocupação de células independentes, relaciona-se com o risco da rede produzir saídas sem significado. Na prática, esse tipo de situação não ocorria com muita frequência (até porque o conjunto de treino não tinha nenhum exemplo disso mesmo), mas mesmo assim era inaceitável que tal acontecesse. Um exemplo desse tipo de saídas está ilustrado na figura 72.

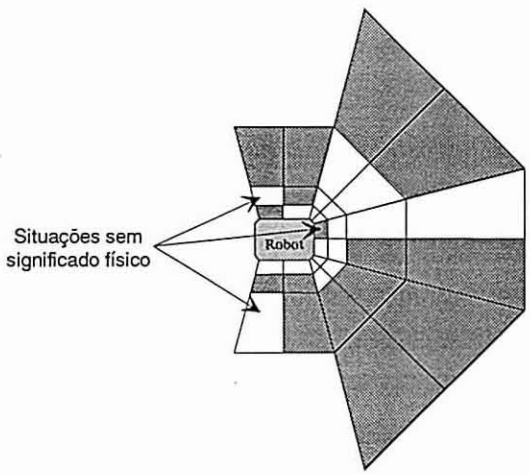


Fig. 72 - Mapa com exemplos de casos de ocupação de células sem significado físico.

A uma célula ocupada devem seguir-se (alinhadas radialmente) apenas células ocupadas. Esta forma de formular o problema com a rede neuronal não assegura que esse princípio esteja garantido.

3.3.2.4. Conclusões sobre uso de redes contínuas

As conclusões neste ponto indicam que o princípio de usar redes neuronais promete sucesso na construção dos mapas de percepção, mas os dados para o seu treino devem ser bem filtrados (e/ou gerados) de forma a eliminar informação contraditória. Essa informação contraditória ocorre devido a intermitências originadas pelas reflexões especulares, sendo muito mais notórias em subconjuntos de treino pequenos.

É necessário subdividir o problema em redes de dimensões menores (que uma rede global) para evitar um universo de possibilidades tão grande que se torna impraticável obter

amostras representativas desse universo (devido também à enorme quantidade de dados necessários).

Uma última conclusão é que não se pode correr o risco de permitir que uma rede neuronal gere mapas sem significado físico, ou seja mapas sobre os quais deverá recair uma análise posterior para eliminar essas situações, e ainda determinar com que critérios fazê-lo, porque pode não ser claro qual o procedimento em face a determinadas combinações de células ocupadas de forma errática. Apesar de ser uma situação raramente encontrada, é forçoso eliminá-la.

Destas conclusões tira-se a necessidade de reformular o problema em termos do uso de redes neuronais: a ideia é a de usar as redes como **classificadores de padrões de ocupação**, como se verá de seguida.

3.3.3. Redes classificadoras de padrões de ocupação

Como foi já adiantado na **Parte 2**, e em virtude das conclusões da secção anterior, recorreu-se a uma forma alternativa de utilizar as redes neuronais. Na abordagem anterior, as redes eram de igual modo classificadores de padrões, mas o que eles representavam era diferente: eram valores individuais de ocupação celular. O que se propõe agora é redefinir as saídas que a rede deverá gerar. Nesta nova formulação tem-se uma representação final mais simples e mais limitada quanto ao número de padrões possíveis (classes) que a rede terá de aprender a reconhecer. O problema fica assim confinado e simplifica-se aparentemente a tarefa da rede, ou pelo menos garante-se uma maior confiança (e controlo) nos valores de saída. Os critérios de convergência são, apesar da reformulação, semelhantes. O erro médio das saídas de todos os padrões a aprender é dado como indicador da evolução da convergência, mas é o número mínimo de padrões com um erro máximo que determina quando uma rede convergiu.

Levadas em conta as conclusões acerca do carácter contraditório de alguns dados sensoriais, foi decidido logo à partida não tentar aprendizagens com esses dados em bruto. Os conjuntos de treino deveriam ser então resultado de uma filtragem dos dados reais aos quais se juntariam dados sintetizados para cobrir os dados dificilmente adquiríveis em algumas situações práticas. Os padrões de treino são pares de vectores **dados/classe de ocupação** com a seguinte formatação:

dados $S = (s_0, ..., s_{N-1})$ e classe de ocupação $C = (c_0,...,c_{M-1})$

com $N=3$ e $M=25$ ou $M=16$ para as redes parciais menores.

$s_j \in [0,+1[$ *e, a classe de ocupação tem todos os membros a zero excepto um que é igual a 1.*

3.3.3.1. Cones (trapezóides) para um mapa global

Os únicos tipos de redes experimentadas com esta abordagem dos padrões de ocupação foram os cones individuais (trapezóides) que, relembre-se, surgem em 3 tipos diferentes. O

motivo é que regiões maiores que os cones implicam um maior número de padrões que a rede terá de aprender a classificar. Imagine-se, por exemplo, que se pretendia usar toda a região frontal (os 3 cones) do mapa global e que seria coberta por 9 sensores, como aliás foi feito no caso de redes como interpoladores contínuos; como essas 3 pequenas regiões têm o mesmo número de células, ter-se-ia um conjunto com 25×25×25=15625 padrões, que é realmente muito se se esperar ter 4 ou 5 pares de treino por cada classe (padrão).

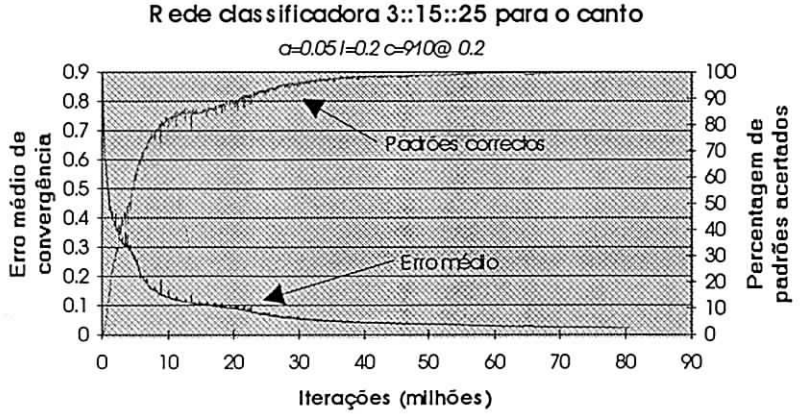


Fig. 73 - Curva de convergência para uma rede classificadora 3::15::25

Foram tentadas redes com camadas escondidas de tamanhos variados, começando por camadas com poucos elementos, e ir aumentando-os até se obterem boas convergências. O resultado final foram redes 3::15::25 para as regiões de 8 células e 3::10::16 para as regiões com 6 células.

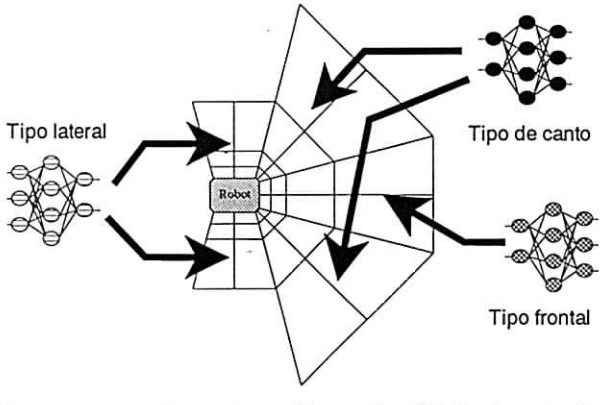


Fig. 74 - A construção de uma mapa de maiores dimensões é feita à custa de várias redes neuronais independentes

A construção do mapa global é feita pela agregação dos diferentes submapas calculados por 3 tipos diferentes de redes; a figura 74 ilustra esse processo. As redes implementadas em *software* consistem na simples multiplicação de matrizes e aplicação de uma função de activação nas saídas dos neurões das diversas camadas da rede. Essa função é muito frequentemente de cariz exponencial. As implicações em termos de custos computacionais são ilustradas de seguida: para um mapa mínimo de navegação (frente e lados) com 5 cones, e

sabendo que o produto de matrizes $[n, m] \times [m, q]$ contém $m \times n \times q$ multiplicações e $(m-1) \times n \times q$ adições, tem-se que as operações aritméticas necessárias para implementar o cálculo do mapa são as seguintes:

$5 \times 3 = 15$ divisões em vírgula flutuante para normalização dos dados sensoriais.

3 vezes:

$[1,3] \times [3,15] \rightarrow$ 45 multiplicações e 30 somas,

$[1,15] \times [15,25] \rightarrow$ 375 multiplicações e 350 somas,

2 vezes:

$[1,3] \times [3,10] \rightarrow$ 30 multiplicações e 20 somas,

$[1,10] \times [10,25] \rightarrow$ 250 multiplicações e 240 somas,

num total de 1820 multiplicações e 1660 adições em vírgula flutuante.

3.3.3.2. Conclusões

Estas redes classificadoras apresentaram capacidades de generalização muito elevadas, como se verificou na prática (tabela 3). Funcionam tão bem que muitas reflexões especulares (de um sensor, essencialmente) passam despercebidas. O sistema parece realmente conseguir aproveitar a redundância espacial dos sensores duma forma mais eficiente e melhor do que os processos algorítmicos resultantes do método alternativo que se descreverá adiante.

	Redes Interpoladoras	**Redes Classificadoras**
Rede **global** para um mapa de percepção com **8** direcções. **24** sensores para **60** células	Taxas de reconhecimento entre **60 e 100%** com dependência da proximidade ao ambiente.	*Não testado (demasiadas classes)*
Rede **parcial** para mapa com **5** direcções e dados **reais. 15** sensores para **40** células	Idem, acrescido de **maior dificuldade** de convergência das redes.	*Não testado (demasiadas classes)*
Rede **parcial** para mapa com **3** direcções e dados **reais. 3** sensores para **6 ou 8** células	**A rede não** convergiu: aprendeu ≈75% dos padrões. **Inconsistências** estatísticas dos dados.	**Não** convergiu. Codificação da ocupação das células complicou a tarefa de -aprendizagem da rede.
Rede **parcial** para mapa com **3** direcções e dados **sintetizados. 3** sensores para **6 ou 8** células.	Convergência **muito boa** (até **99%** dos padrões) e bastante rápida (poucas épocas até aos **80%** de convergência) *(Não testado no robot)*	Convergência **muito boa** (até **99%** dos padrões) Taxas de reconhecimento **superiores a 95 %** mesmo quando com **1** sensor em **reflexão especular.** Taxas menores (< **60%**) com **2** sensores em reflexão **especular.**

Tabela 3 - Comparação das eficácias das mais importantes redes experimentadas.

A abordagem foi relativamente simples no sentido em que os sensores são analisados (processados) em pequenos grupos (3 de uma vez). O princípio poderá eventualmente ser

estendido, e mais sensores poderão ser levados em conta simultaneamente de forma a gerar submapas maiores e mais robustos aos dados sensoriais faltosos.

3.4. Construção dos mapas de percepção

Os mapas gerados com o método das redes neuronais como classificadores mostraram-se adequados e permitiram a navegação, como se verá mais tarde. É contudo pertinente questionar eventuais alternativas para a sua determinação, pelo menos para comparar as vantagens ou desvantagens dos métodos. Foi isso que foi feito, e de seguida ilustram-se os resultados com outros dois métodos, um dos quais levado a cabo por ser de implementação algo simples, mas também simplista como se verá.

3.4.1. Resultados com os métodos alternativos

Na **Parte 2** foram descritos os princípios de dois métodos alternativos às redes para o cálculo dos mapas de percepção (método IPEL, e método baseado em regras). Como se mostra adiante, os resultados não foram maus, mas ficou provado, tal como foi dito, que nenhum dos dois métodos satisfez como o método das redes neuronais cujas características (vantagens e desvantagens) principais se podem listar do seguinte modo:

- As redes neuronais são treinadas para uma geometria em particular. A fase de treino pode ser relativamente longa, mas só tem se ser feita uma só vez.
- As redes neuronais são propícias a modificações e melhorias no sistema:
  - Adicionar ou tirar entradas para o cálculo dos mapas (p/ex. mais sensores)
  - A sua arquitectura permite a introdução de *feedback* se isso se mostrar necessário.
- As redes neuronais tendem a providenciar acções de filtragem no sentido em que entradas muito diferentes das aprendidas são feitas corresponder a um padrão aprendido cuja entrada é o mais semelhante possível à entrada corrente. Como exemplo, alguns casos de reflexões especulares não resultarão em padrões de saída estranhos uma vez que a rede foi treinada para um conjunto restrito de reflexões especulares. Desse modo, uma situação muito diferente das ensinadas será aproximada por uma situação o mais semelhante possível dentro do conjunto de treino, é no fundo a capacidade de generalização.
- Além da utilização como puros classificadores na acepção explicada anteriormente, as redes neuronais como sistemas interpoladores poderão ter aplicação na geração de valores numéricos contínuos de ocupação que poderão ser interpretados ou integrados noutros sistemas para quem esses valores contínuos sejam importantes para interpretações dinâmicas dos mapas, por exemplo.

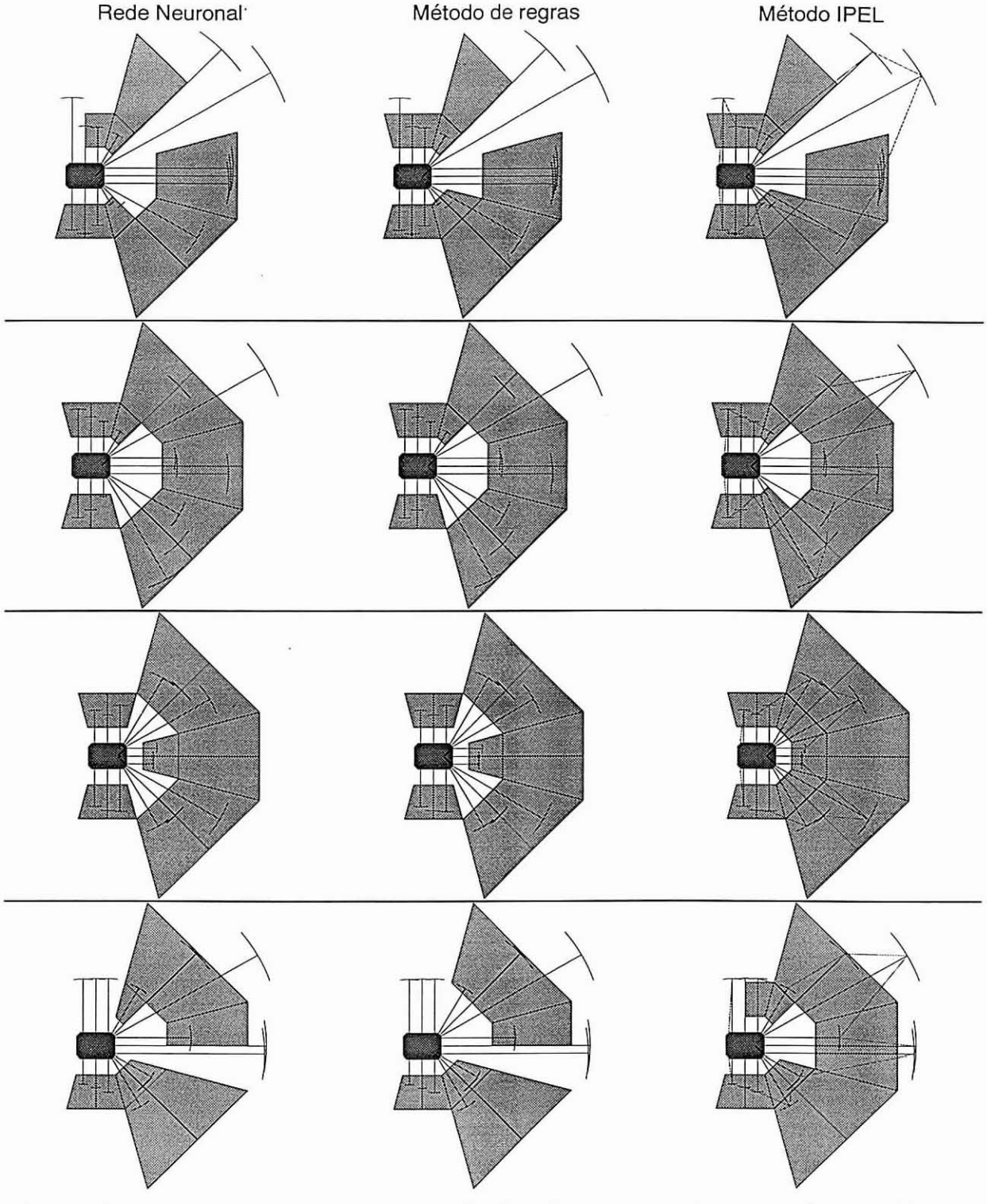


Fig. 75 - Quatro exemplos de Mapas de Percepção calculados por uma rede neuronal (à esquerda), pelo método baseado em regras (ao meio) e pelo método IPEL (à direita)

As características do método IPEL foram já enumeradas na **Parte 2**. O seu uso não foi experimentado no robot pelo motivo de ser mais trabalhoso de implementar nessa plataforma. Isso deve-se também ao facto de requerer uma representação analítica da grelha do mapa para poder calcular as intersecções, e de isso ser computacionalmente intenso. Vistas ainda as

outras desvantagens, não era de facto muito pertinente experimentar além da simulação na estação de trabalho.

O método baseado em regras simples para a construção dos mapas como descrito na **Parte 2** foi de facto implementado no robot. A sua implementação é relativamente simples uma vez que se trata de uma hierarquia de regras de comparação de medidas sensoriais com as diversas dimensões celulares.

O método é perfeitamente algorítmico. Os mapas que gera são, em termos práticos, de qualidade razoável, como se verificou. Apresenta contudo uma grande desvantagem: os mapas variam à taxa directa dos dados sensoriais. Em circunstâncias de leituras intermitentes ou instáveis de um ou mais sensores os mapas assim podem vir intermitentes e instáveis. Este tipo de fenómeno raramente se verificava nas redes neuronais quando um só sensor apresentava esse comportamento de instabilidade.

Outra desvantagem deste método baseado em regras de comparação é não ser facilmente modificável se se quiser inserir mais sensores como já foi indicado anteriormente. Ter-se-ia que rescrever a regras, e o conjunto dessas regras tenderia a complicar-se de forma a prever todas as interacções entre si: um conjunto de regras para 3 sensores e 8 células será de certo mais simples do que um conjunto de regras para 5 sensores e 16 células, por exemplo. Com uma rede neuronal esse problema não se afigura tão grave, se bem que não tenha sido experimentado.

	Rede Neuronal	**Método baseado em regras**	**Método IPEL**
Princípios do cálculo	Multiplicação de Matrizes.	Regras de comparação de combinações de medidas.	Teste de intersecção de múltiplas linhas poligonais.
Numero de mapas por segundo no sistema do robot	2.5	> 10	*Não testado*; mas, por comparação noutro ambiente, inferior à rede.
Sensibilidade às falsas medições	Elimina cerca de 90% das reflexões especulares de um só sensor. É portanto muito insensível a falhas sensoriais.	Reage instantaneamente a todas as variações sensoriais em função das regras definidas.	Reage a todas as variações sensoriais. Um sensor em falha permanente pode inviabilizar os mapas.
Particularidades de implementação	A rede é treinada para uma configuração específica de sensores.	Dificuldade em definir as regras todas, especialmente para as situações anómalas de medição (falhas)	Produz mapas sobreocupados desperdiçando espaço livre.
Operações para calcular um mapa excluindo normalização de dados	1820 multiplicações 1660 adições	Cadeia de comparações numéricas até um máximo de 5×3×4=60 para um número mínimo de regras.	Até 160×15=2400 testes de intersecção de segmentos

Tabela 4 - Resumo da comparação dos 3 métodos para calcular os mapas de percepção.

A figura 75 ilustra mapas de ocupação gerados por uma rede neuronal, pelo método baseado em regras de comparação e pelo método IPEL. Compare-se os resultados e verifica-se que a rede "aproveita" melhor o espaço livre que o método IPEL por exemplo. A vantagem das redes reside sobretudo na dinâmica da geração dos mapas, onde fornece uma maior continuidade entre mapas consecutivos, ou oriundos de algumas situações de intermitência

sensorial. A grande vantagem das redes relativamente aos outros métodos é a sua robustez (insensibilidade) às falsas medições. Em termos de exigências computacionais o método das regras é muito menos exigente, mas também não apresenta a robustez desejada às falsas medições. A tabela 4 resume as principais características dos 3 métodos.

3.5. Estratégias de navegação — o algoritmo

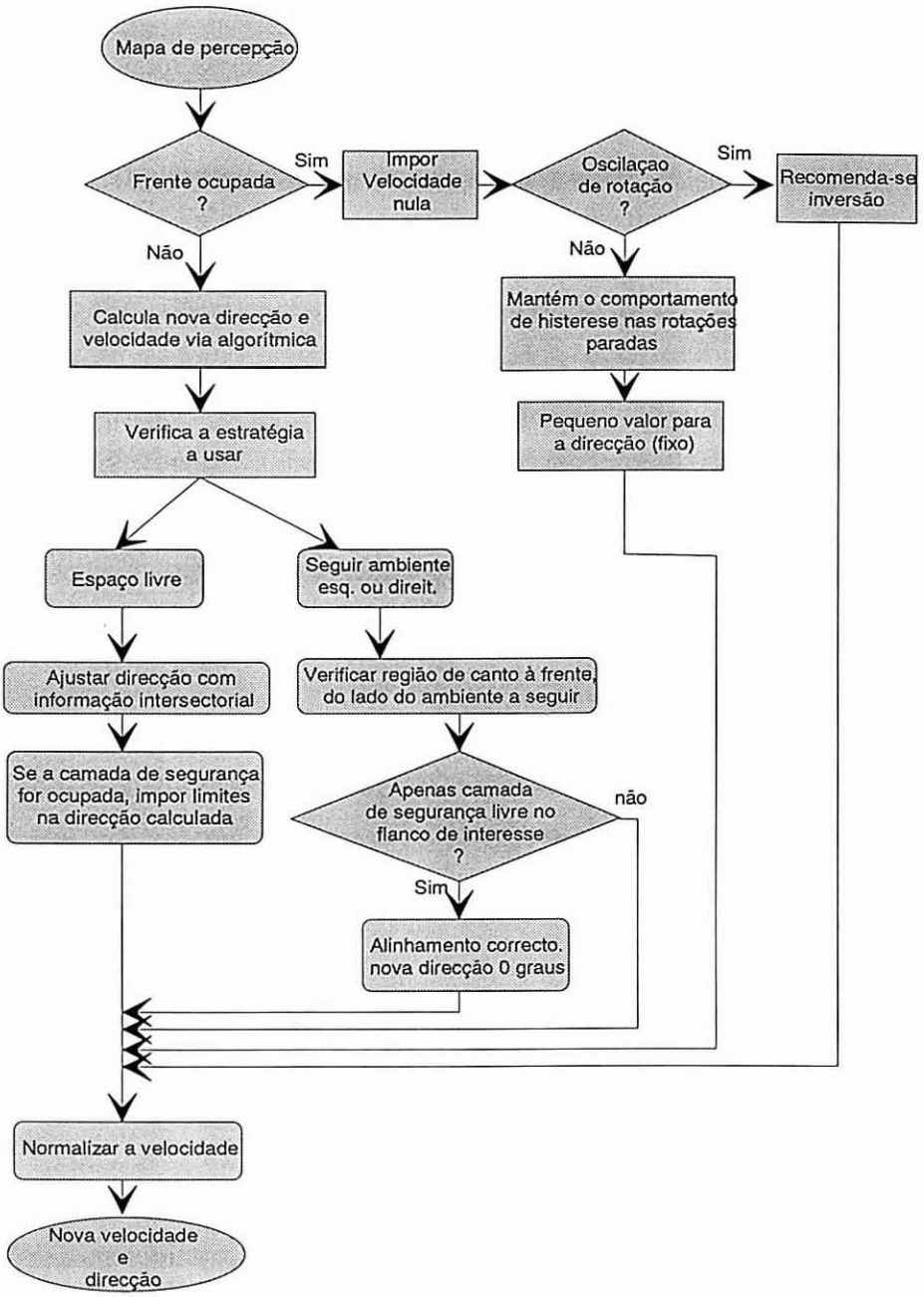


Fig. 76 - Diagrama simplificado do algoritmo da determinação da nova velocidade e direcção do robot na navegação local.

Conforme foi dito na **Parte 2**, um estratégia tem por função determinar a cada instante a nova direcção e velocidade do robot mantendo uma determinada linha de comportamento e em função do espaço livre indicado pelo mapa de percepção nesse instante. Foram derivadas e

explicadas as fórmulas para o cálculo destas grandezas para um caso geral, mas foi logo antevista a necessidade de complementar essas directivas numéricas com alguma análise lógica para situações especiais de navegação, como se descreverá abaixo.

3.5.1. A componente numérica do algoritmo de navegação

A componente numérica é aquela que calcula valores numéricos para a nova direcção e velocidade usando as fórmulas apresentadas e os coeficientes adequados. Em muitas circunstâncias estes valores chegam não perturbados ao termo do algoritmo: depende de estratégia e sobretudo da proximidade do robot ao ambiente num dado momento. O único ajuste que se faz sempre é uma normalização e limitação da velocidade final, fundamentalmente por motivos de segurança.

A optimização dos parâmetros das estratégias (coeficientes ponderadores internos de uma região e coeficientes ponderados entre as diversas regiões) é um processo complicado dado o grande número de variáveis independentes, como se viu na **Parte 2.** Algumas tentativas de optimização teórica foram feitas, mas a experimentação revelou-se mais rápida nos resultados, principalmente porque se tinha concebido um sistema de simulação antes de passar à prática, como se verá mais à frente.

3.5.2. As condições especiais no algoritmo de navegação

As **condições especiais** para a determinação da nova direcção e velocidade são aquelas para as quais o recurso à parametrização (fórmulas) das estratégias é insuficiente ou a determinação de parâmetros seria muito complexa. São portanto situações em que a nova velocidade e direcção do robot são determinadas de outra forma que não a expressão aritmética. Conforme também já descrito na **Parte 2,** as **condições especiais** incluem as seguintes situações:

- Desequilíbrio de ocupação celular nos dois sectores de uma região (direcção fundamental).
- Ocupação da camada de proximidade mínima (segurança) do mapa de percepção.
- A oscilação de rotação em posição parada (velocidade nula).
- Situação de bloqueio (*deadlock*) e consequente inversão de marcha.
- Procedimentos após as paragens de emergência.

As duas primeiras situações dizem respeito a uma análise suplementar do mapa de percepção. Independentemente do cálculo ponderado de células ocupadas e livres verificar se, no mapa corrente, a camada mais próxima do robot já tem células ocupadas, é necessário tomar cuidados extra. O mesmo é válido no desequilíbrio entre sectores da região frontal sobretudo, como foi exemplificado na **Parte 2.**

Verificar se se entrou em oscilação de rotações, situação muito provável junto a um canto duma sala, é determinante para se poder suplantar as directivas numéricas e impôr ao

veiculo histerese na direcção de rotação. Dessa forma o veículo acabará por reencontrar o espaço livre em frente e deixará de rodar em torno de um eixo fixo: o ciclo de histerese cessa até ao próximo canto de parede!

A situação de bloqueio ocorre quando não se pode ir para a frente e se está em oscilação de pequenas rotações. Nessa circunstância, e mais uma vez pela simetria do sistema (robot e mapas) proceder a uma inversão é a solução: a frente passa a ser retaguarda e vice-versa. A tabela de sensores é transformada e os módulos de navegação local não se apercebem de nada. Na verdade há ligeiras diferenças porque a dinâmica do robot não é a mesma: a tracção atrás ou à frente resulta num comportamento mecânico ligeiramente diferente, devido ao eixo tractor não estar centrado no próprio robot.

Determinados tipos de navegação podiam pecar por superabundância de situações especiais levando a um comportamento menos contínuo e afirmado do robot: era o caso de espaços de movimentação muito apertados ou ambientes muito dinâmicos nas vizinhanças (como pessoas a circular muito e a colocar obstáculos propositadamente na sua trajectória).

A consequência deste misto de algoritmia e multiplicidade de situações especiais resultava num sistema não muito determinístico, isto é, nunca se conseguia prever para que lado virava o robot ao chegar a uma parede ao fundo da sala.

Das condição especiais listadas anteriormente faltou mencionar os procedimentos após as paragens de emergência. Na realidade dever-se-ia dizer: após um grande número de paragens de emergência consecutivas! Esta condição especial pode resultar de uma luta entre o detector de emergências (que ganha sempre) e o navegador local. Uma situação que ocorria frequentemente está ilustrada de seguida:

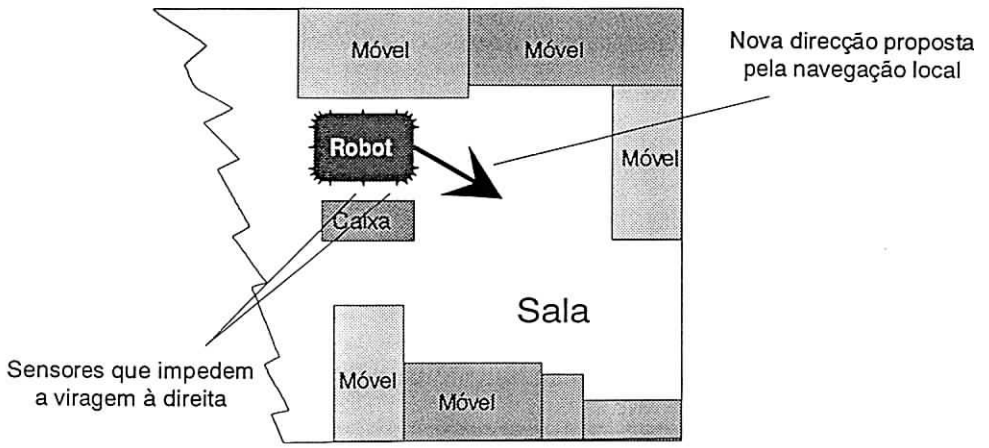


Fig. 77 - Situação de luta entre o navegador local que propõe uma direcção e o detector de emergências.

Na situação da figura 77 é necessário evitar a situação de "luta" que ocorreria se apenas se usasse o algoritmo básico descrito na figura 76. Para esta situação criou-se um pequeno bloco que fazia avançar o robot lentamente para frente (se houvesse espaço livre em frente) durante poucos segundos e curto-circuitando por completo a informação vinda do algoritmo base da estratégia de navegação. Ao fim desse tempo em que o robot teria andando 15 ou 20

cm, a informação do algoritmo base é de novo considerada e o sistema retoma a via normal ou entra ainda numa fase destas até conseguir sair daquela situação de aperto. Na prática nunca se constataram mais de 2 ou 3 sequências destas acções auxiliares.

3.6. A simulação do sistema de navegação local

A necessidade de simulação do sistema antes de o implementar na prática era mais que óbvia. Tratando-se de trabalho inédito em termos de autonomia deste robot, seria no mínimo prudente tentar prever qual o comportamento do robot sob a acção de estratégias de navegação nunca testadas sobre mapas de percepção, que apesar de resultarem de redes neuronais bem treinadas (por boas condições de convergência na aprendizagem), não garantiam necessariamente uma grande robustez à dinâmica do robot. Esta prudência é justificada pelas questões de segurança associadas às dimensões e peso do robot bem como a força dos motores de tracção!

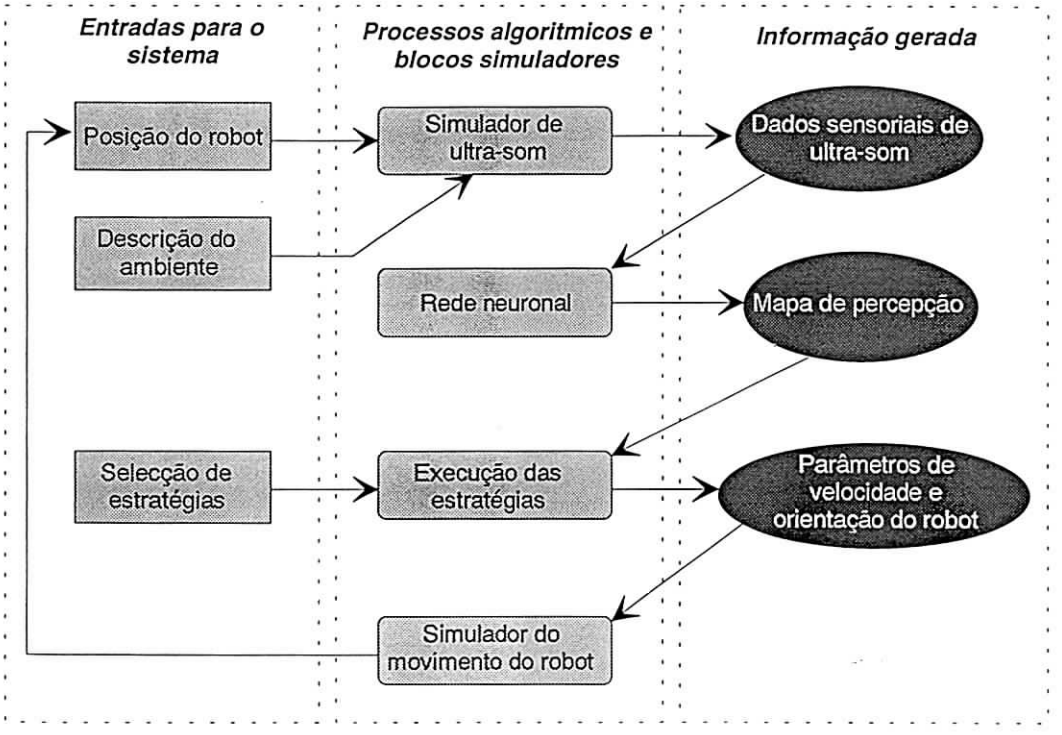


Fig. 78 - Esquema da simulação da Navegação Local para o robot

Para o estudo na navegação local usando mapas de percepção construídos com redes neuronais a partir de dados de sensores de ultra-som foi preciso simular os seguintes subsistemas:

- Simulação dos dados dos sensores de ultra-som
- Simulação do movimento do robot

Além destes componentes, e como se pode ver na figura 78, outros elementos fazem parte do esquema de simulação, mas aí não se trata de simulação: é a implementação ela

própria. É o caso das estratégias de navegação e das redes neuronais. O ambiente de simulação foi em *Sun SparcStation* e o sistema real do robot é ambiente *Motorola 680x0*.

A simulação dos dados de ultra-som levou em conta alguns factores associados aos sensores, mas não cobriu todos os aspectos da medição com ultra-som (ver Apêndice). Os valores sensoriais simulados não se afastam muito de dados ideais, o que todavia não é ilícito por se tratar da aproximação à situação desejada.

3.6.1. A simulação do movimento do robot

A simulação do movimento do robot foi talvez, de todas as componentes da simulação empreendida, a menos fiel à realidade. O que se pretendia era dar a cada instante a nova posição do robot. A simulação não tomou em conta os momentos de aceleração/desaceleração, mas manteve a continuidade da trajectória gerada e obedeceu às contingências de ordem holonómica: o robot não andava de lado, por exemplo!

O posição do robot era actualizada em intervalos de tempo regulares, tendo em conta a orientação e a velocidade com que tinha sido deixado no final do intervalo de tempo anterior: a actualização da posição era assim de cariz inercial. Isto não é um problema grave se estes intervalos de tempo forem muito menores que os intervalos entre os quais há mudança da velocidade ou da orientação do robot, o que se procurou fazer.

A simulação do movimento era igualmente a de um "movimento perfeito", isto é não tomava em conta quaisquer irregularidades na aderência e contacto com o solo, o que na prática é errado, mas que não afecta as conclusões desta tarefa, porque tratando-se navegação local não interessa ter nenhum referencial solidário ao ambiente ou pavimento.

3.6.2. Resultados do simulador e seus problemas e limitações

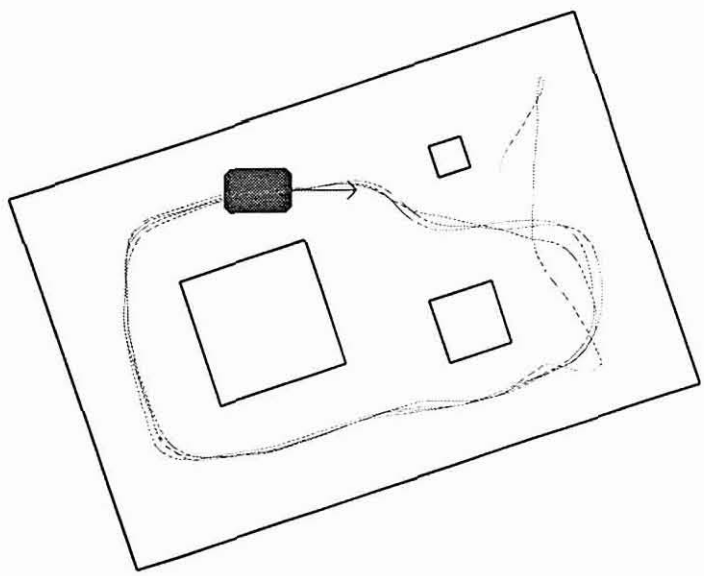


Fig. 79 - Simulação de uma trajectória usando uma estratégia de seguir o espaço livre.

Apesar das aproximações com que o simulador foi construído, os resultados mostraram-se interessantes. A figura 79 mostra uma trajectória resultante de uma simulação usando uma estratégia de seguir o espaço livre.

Apesar deste e doutros resultados prometedores para a implementação prática, o simulador apresentava as suas limitações, quer inerentes à concepção, quer no desempenho realista das suas funções. Os principais problemas e limitações da simulação resumem-se do seguinte modo: a simulação tornava-se computacionalmente muito intensa para ambientes muito complicados (muito elementos). Não previa também situações de paragem de emergência, logo não dispunha de procedimentos para sair dessa situações. A simulação não estava igualmente preparada para proceder à inversão de marcha aquando de situações de bloqueio (*deadlock*); detectava a necessidade de inverter a marcha mas não simulava o robot nesse modo. Todavia estas situações são tão especiais que não impedem a validade do simulador. Na realidade foi imprescindível para determinar os parâmetros e coeficientes para o algoritmo de navegação.

3.7. Execução das estratégias de navegação

Nesta secção mostram-se resultados reais do comportamento do robot na execução das estratégias de navegação. O robot era simplesmente entregue a si próprio: a estratégia era imposta pelo utilizador e eram as ocorrências de ambiente que moldavam o comportamento do robot dentro do espírito de cada estratégia. É um pouco difícil ilustrar a dinâmica do veículo recorrendo a imagens estáticas, por isso é de aconselhar o visionamento da gravação vídeo referente a estes resultados de navegação.

3.7.1. Navegação com estratégia seguindo os espaços livres

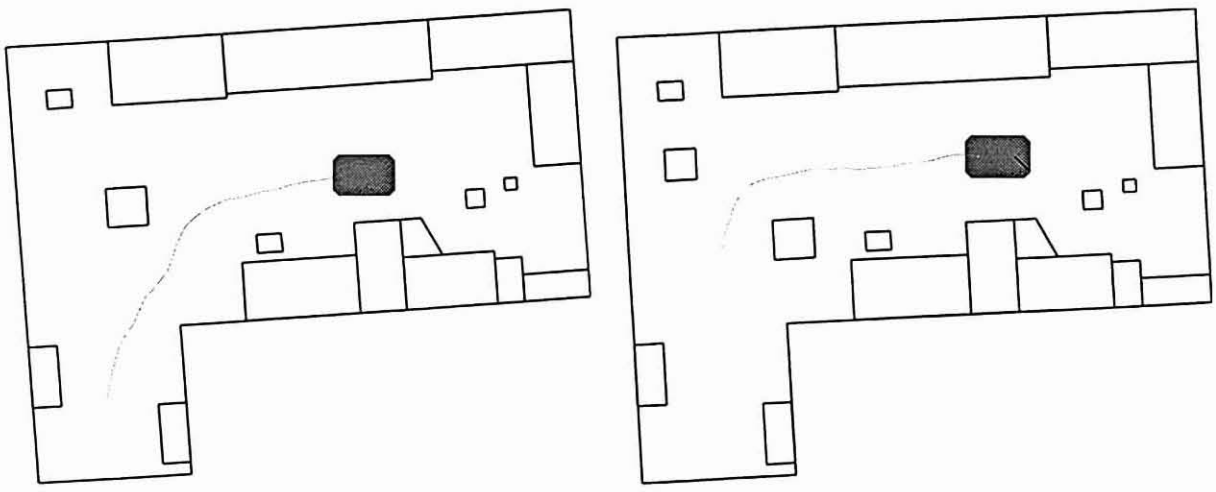


Fig. 80 - Trajectórias reais executadas pelo robot usando uma estratégia de seguir o espaço livre. Em ambos os casos o robot partiu do mesmo ponto, mas o ambiente foi modificado pelo deslocamento de uma caixa colocada no trajecto.

A estratégia de seguir o espaço livre foi a primeira a ser implementada, e também aquela cujos parâmetros foram mais fáceis de determinar. O facto de esta estratégia fazer uma ponderação perfeitamente simétrica dos elementos do mapa de percepção facilitou essa tarefa de ajuste desses factores.

A figura 80 dá dois exemplos de trajectórias resultantes, das inúmeras executadas, usando a estratégia de seguir o espaço livre.

3.7.2. Navegação com estratégia seguindo o ambiente

As estratégias de seguir o ambiente são duas, mas na verdade estão muito relacionadas. Devido à simetria física do robot, seguir o ambiente no flanco esquerdo ou no flanco direito é indiferente em termos conceptuais. As estratégias de seguir o ambiente foram mais difíceis de ajustar no que respeita aos parâmetros e coeficientes. De seguida ilustra-se a trajectória real resultante de se usar uma estratégia de seguir o ambiente existente no flanco direito do robot, preservando distâncias de segurança como a estratégia impõe.

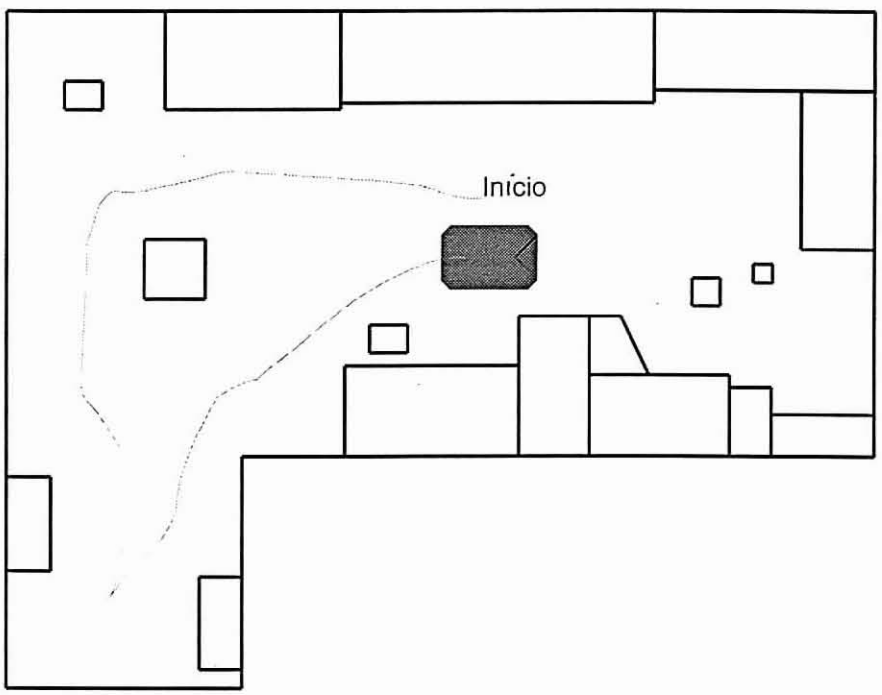


Fig. 81 - Trajectória real usando a estratégia de seguir o ambiente à direita do robot. Note-se a volta executada a cerca de metade da trajectória por falta de espaço livre em frente.

3.7.3. Navegação com estratégias mistas em sequência

O caso mais geral de movimentação do robot é utilizar combinações várias de estratégias de navegação em sequência. Na prática isso foi experimentado: a selecção das estratégias foi feita manualmente, mas nada mais era dito ao robot. Um exemplo concreto, realmente realizado, destas possibilidades vem ilustrado na figura 82. Aí pode-se ver que o robot foi inicialmente "lançado" usando uma estratégia de seguir o espaço livre; a certa altura, para garantir que seguiria pela esquerda do obstáculo, impôs-se temporariamente uma

estratégia de seguir o ambiente à direita, e pouco depois uma estratégia de seguir o ambiente à esquerda garante um ciclo infinito em torno do obstáculo central. Para sair desse ciclo, é-lhe oportunamente colocada uma estratégia de seguir o espaço livre e o efeito é o indicado.

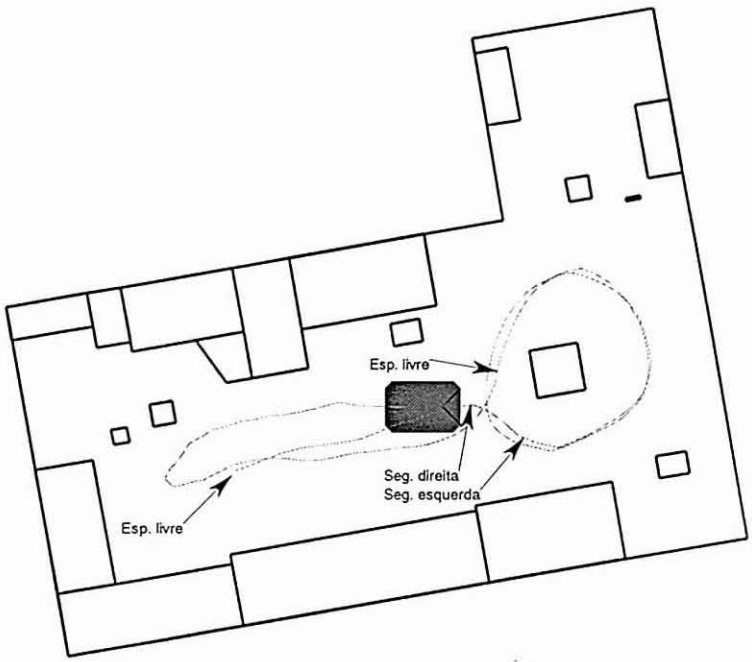


Fig. 82 - Exemplo de uma trajectória resultante de combinação de várias estratégias de navegação em sequência. Na figura estão indicados os pontos aproximados onde a nova estratégia entrou em acção.

3.8. Conclusões

As conclusões deste trabalho foram divididas em várias categorias:

- A validade dos conceitos introduzidos.
- O sucesso experimental das técnicas propostas.
- A possibilidade de generalização para outros sistemas.
- O grau de resolução dos problemas propostos ao início.

3.8.1. Validade dos conceitos introduzidos

Os conceitos introduzidos, ou profundamente readaptados, provaram a sua utilidade e validade com os resultados práticos descritos atrás. Eis de seguida um resumo de cada um dos mais importantes:

3.8.1.1. Grelhas de Geometria Geral - GGG

As grelhas de geometria geral foram introduzidas e definidas de uma forma algo formal, pela simples razão porque foi essa a sua forma de aparecimento: todas as suas propriedades surgiram como necessárias para o problema em causa. Pode-se dizer que foram as necessidades de um problema concreto que levou ao formalismo e às regras expostas. No fundo, são estruturas geométricas que pretendem generalizar o conceito de grelha, nomeadamente de grelha cartesiana. A forma como foram criadas foi em resposta a um tipo de

problema e determinadas considerações práticas e de implementação. As limitações incluem o facto de não permitirem a definição de forma fácil todas as grelhas possíveis, sobretudo devido à condição de recursividade na geração sucessiva de células menores, e também não permitem a definição de grelhas de dimensões infinitas, mas essas questões são irrelevantes para a aplicação onde se mostraram aliás muito úteis.

3.8.1.2. Mapas de Percepção

A expressão "Mapa de Percepção" parece tão óbvia que o seu uso não é por certo novo. Encontram-se as expressões **Mapas Locais, Mapas Globais, Mapas Topológicos, Mapas Geométricos,** mas nenhum se identifica bem com a acepção aqui explorada. É claro que estes mapas de percepção são locais, mas são-no como resultado exclusivo de dados sensoriais, na linha de vista e mudam continuamente em conformidade com a chegada de dados sensoriais frescos. O mapa de percepção traduz uma propriedade do espaço que os sensores captam; neste caso foi a ocupação desse mesmo espaço. O conceito pode alargar-se e definirem-se mapas, por exemplo, de temperatura, ou densidade de gases, ou intensidade de radiação, uma vez encontrados os sensores que os permitissem determinar. Desta forma, a validade deste conceito não parece questionável, neste caso em especial porque não são mais do que uma qualificação acrescentada a uma estrutura que já anteriormente se discutiu ser válida: as GGG.

3.8.1.3. Navegação Local

A expressão "Navegação Local", tal como explicado nos capítulos da **Parte 2,** já existia, mas com um significado diferente daquele que lhe foi dado ao longo deste trabalho. A navegação tornou-se realmente local, no sentido em que é executada tendo em conta apenas informação local (próxima) do ambiente (a percebida pelos sensores) e não qualquer conhecimento sobre o destino final ou sobre o restante ambiente. Tem-se portanto que esta nova definição tem uma validade muito grande porque se trata da introdução de um paradigma novo de navegação que conseguiu resolver os problemas a que se propunha, conforme se verificou com os resultadas experimentais...

3.8.1.4. Estratégia de Navegação

A expressão "Estratégia de Navegação" traduz um conceito primordial na navegação local na acepção introduzida neste trabalho. Foi definida essencialmente como uma norma de comportamento de teor muito qualitativo, um pouco à semelhança das acções humanas. A estratégia é a acção a partir da percepção (obtida com os sensores). Esta acção é por vezes pouco determinística ao nível macroscópico. A diversidade de estratégias não é muito grande (duas fundamentais foram expostas) tal como se poderia esperar dada a dificuldade em predizer exactamente as suas consequências, acima de tudo porque são reacções a um conjunto de fenómenos muito imprevisíveis ou difíceis de quantificar: dados sensoriais por vezes faltosos, interacção mecânica do robot com o pavimento ou o ambiente em geral, etc.

Tem-se assim que a estratégia de navegação é o elemento necessário e responsável último em consciência pela acção (movimento) da própria navegação, fazendo dela uma concepção indubitavelmente válida.

3.8.1.5. Arquitectura Global de Navegação

Concluir sobre a validade da arquitectura proposta é bastante mais difícil, até porque a arquitectura não foi toda implementada na prática, e alguns módulos poderão ter de sofrer uma reformulação por motivos práticos ou de implementação. Todavia, as secções que foram implementadas mostraram-se eficientes. A arquitectura proposta segue as tendências que se começam a encontrar no que diz respeito à modularidade e reactividade. São de relembrar os 3 ciclos fundamentais situados a vários níveis: um que se fecha pela detecção de colisão e consequente paragem do veiculo (constitui a manobra de recurso); um ciclo de maior alcance que inclui a navegação local e que permite o movimento mesmo sem trajectórias pré-definidas baseando-se simplesmente nos dados de ultra-som (permite o movimento seguro), e finalmente um ciclo de alcance extremo que corre a maior parte dos módulos funcionais e que permitirá, quando completamente desenvolvido, tarefas completas de navegação.

Deixaram-se também ideias para implementar outros módulos em particular o seguidor de trajectórias. Os desenvolvimentos futuros o dirão se esta arquitectura foi uma concepção válida ou não.

3.8.2. Sucesso experimental das técnicas propostas

Os mapas de percepção são um conceito que representa de forma útil e bem sucedida a possibilidade de representar o espaço livre nas imediações do robot. As representações obtidas funcionaram bastante bem.

A técnica das redes neuronais deu muito bons resultados na construção dos mapas de percepção. Foi uma técnica inovadora que conseguiu lidar com dados reais ruidosos e permitiu a navegação em tempo real. Foi sem dúvida um dos resultados mais significativos deste trabalho.

A navegação local aparece como uma boa aposta na modularização de uma arquitectura para navegação de um robot móvel. É assim permitida um mais fácil adaptação ou modificação de componentes de navegação mantendo mesmo intocadas as restantes componentes. Foi esta propriedade que permitiu refinar parâmetros de navegação, tentar alternativas a métodos de cálculo de percepção, etc, um de cada vez ou todos simultâneos em qualquer fase de evolução do projecto.

3.8.3. Possibilidades de generalização para outros sistemas

A proposta de navegação feita tem uma dependência aparente com o sistema, principalmente ao nível dos mapas de percepção. Estes foram construídos para uma dada

distribuição geométrica de sensores: contudo, os princípios de definição da grelha (GGG) que forma o suporte topológico dos mapas são perfeitamente lógicos para outras geometrias.

Não existe, portanto, nenhuma dependência fatal nesta abordagem para com um sistema em particular; mantendo esses mesmos princípios seria possível conceber uma grelha para construir os mapas de percepção de, por exemplo, um robot "circular" (fig. 83), mas onde a distribuição de sensores seguisse uma filosofia semelhante, ou seja procurar criar redundância pela sobreposição espacial. Nem sequer a questão da simetria é vital. A definição das GGG possibilita toda uma variedade de anisotropias e assimetrias.

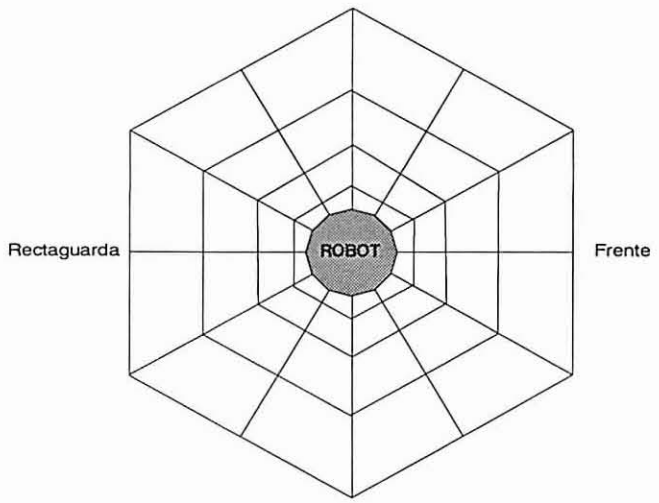


Fig. 83 - Possível grelha para os mapas de percepção de um robot "circular".

Os princípios implementados resumem-se portanto a um fenómeno de integração sensorial e redução da complexidade sensorial. Como consequência surge a definição de descrições menos "finas" da percepção do espaço, mas também mais fiáveis e válidas do que interpretação de sensores isolados.

3.8.4. O grau de resolução dos problemas apresentados

O balanço acerca dos problemas enumerados na **Parte 1** relativos aos sistemas tradicionais e para os quais se propunham avanços para as respectivas soluções é como segue:

3.8.4.1. Necessidade de conhecer a posição corrente para as acções de navegação

A proposta apresentada, desenvolvida, implementada e testada levou à definição de um conceito de navegação onde a posição corrente do robot no ambiente é completamente desconhecida e irrelevante para as acções de navegação local que englobam em particular o desvio de obstáculos.

3.8.4.2. Escassez de técnicas baseadas exclusivamente nos sensores a bordo.

As técnicas e métodos propostos baseiam-se na percepção do ponto de vista do robot apenas, logo recorre exclusivamente ao sensores transportados a bordo do robot e não de

sensores colocados no ambiente: é o próprio ambiente ocorrente que serve de referência à navegação.

3.8.4.3. A maioria das acções de navegação são feitas em função do destino final.

Este problema invoca uma característica dos métodos dos Campos de Potenciais e que foi circunscrita na abordagem levada a cabo neste trabalho. As acções de navegação [local] ocorrem em função do espaço livre para o movimento e de uma tendência geral (comportamento) que não está necessariamente orientada em seguir uma direcção que leve ao destino final. Isso é uma preocupação a outro nível: o sistema desenvolvido pode inclusivamente prescindir de um destino final (deambulação).

3.8.4.4. Poucas representações do espaço livre nas imediações de um robot móvel.

Este problema teve por certo alguns passos importantes neste trabalho: garantidamente não foi dito ou explorado tudo sobre a representação do espaço livre captado pelo robot em torno de si próprio, mas o problema foi abordado para um caso particular de percepção ultra-sónica e pode muito possivelmente ser estendido a outros tipos de percepção. Os mapas de percepção desempenharam aqui um papel central.

3.8.4.5. Dificuldades com os problemas associados a ultra-sons.

Não foram resolvidos todos os problemas da medição de distâncias através de ultra-sons, mas foram desenvolvidas e testadas com sucesso técnicas de minimização dos efeitos nefastos de interferência (*crosstalk*) bem como das reflexões especulares recorrendo à redundância espacial de agrupamentos com múltiplos sensores.

3.8.4.6. Sistema com propriedades reactivas

Esta questão refere-se essencialmente à importância do tipo de arquitecturas de navegação em robótica móvel que se caracterizam pela reactividade em função dos dados sensoriais. Na verdade, a arquitectura completa de navegação sugerida, se bem que não experimentada na sua totalidade, mostrou-se válida na forma como a informação estava organizada. A reactividade do sistema à informação dos sensores é bem clara. O bloco de Navegação Local é um exemplo nítido de sistema reactivo aos sensores. As directivas de que necessita para reagir (produzir movimento) a novos valores de percepção são mínimas e em certos casos até suficientes para uma navegação reduzida mas segura.

3.8.4.7. Aspectos de autonomia

A autonomia não foi uma questão exposta explicitamente, mas era no fundo o problema que se queria abordar. Tratava-se pois de conferir ao sistema (robot móvel) uma capacidade de navegar com segurança num ambiente sobre o qual se tem um conhecimento incompleto (ou

nulo sob o ponto de vista de alguns módulos) e de se adaptar às configurações ambientais de ocupação de espaço de forma a poder cumprir a execução das tarefas de navegação.

De um sistema existente quase exclusivamente manual (ver Apêndice) passou-se a um sistema que toma acções (decisões) em função do ambiente que o circunda sem a intervenção humana. Mais ainda, essas acções são muitas vezes um pouco imprevisíveis nos seus detalhes mas obedecem a um comportamento lógico esperado. Caminhou-se em direcção a um sistema autónomo.

Uma outra aplicação que o sistema permite e que foi em boa parte implementada, reside na chamada **navegação manual assistida.** Ao operador é dada a prioridade no comando dos movimentos do robot. Quando se verificar uma condição especial, como por exemplo um obstáculo na trajectória ou uma qualquer eminência de colisão, o sistema pode fazer mais do que simplesmente parar e acusar a ocorrência: toma ele em mão a navegação seguindo uma estratégia de defeito, e pode por exemplo prosseguir para o meio de um corredor de onde o operador inadvertidamente se tinha desviado. O operador pode em qualquer momento retomar o controlo manual. A facilidade da navegação assistida resulta de um simples acréscimo de protocolo entre os módulos de navegação; é uma outra virtude da arquitectura de navegação proposta (ver **Parte 2**).

3.8.5. Problemas deixados em aberto e possibilidades de melhoria

A solução proposta apresentou excelentes resultados como foi descrito anteriormente. Se alguns problemas ficaram resolvidos ou minimizados, há outros que ficaram a descoberto. Desses destacam-se os seguintes:

- O regresso à trajectória perdida usando navegação local é um problema deixado em aberto.
- Afinação dos parâmetros da navegação para reduzir o número de condições especiais.

O primeiro destes problemas foi de certo modo previsto (ver final da **Parte 2**), e é difícil dizer até que ponto ele é factível ou interessante do ponto de vista exclusivo da navegação local em si. Talvez uma cooperação com um bom sistema de localização e uma extensão dos princípios das estratégias de navegação seja possível rumar a esta ideia de recuperar o desvio de uma trajectória devido à necessidade de, por exemplo ter de evitar um obstáculo inesperado e/ou desconhecido inicialmente.

O segundo problema é um pouco específico à técnica de implementação das estratégias de navegação. Na verdade a abundância de parâmetros torna muito difícil a sua optimização, são sistemas a muitas variáveis em que muitas vezes se recorre a uma "optimização" de relação entre variáveis para tentar reduzir o seu número. Esta questão pode ser resolvida de duas formas: seleccionar parâmetros de menor importância e eliminá-los ou reformular a

implementação das estratégias de navegação, como por exemplo recorrendo a uma rede neuronal ensinada com dados resultantes de acções de navegação humana.

Uma outra questão que não é sequer um problema significativo, mas que poderia trazer alguns benefícios em termos de fluidez e estabilidade de navegação, tem a ver com a granularidade do mapa de percepção. Seriam talvez precisos mapas mais finos com redes ainda mais robustas às reflexões especulares, o que passaria quase necessariamente por colocar mais sensores no sistema.

Parte 4

Apêndices.

Apêndice A - Conceitos sobre Redes Neuronais.

A.1. Introdução

As redes neuronais são sistemas orientadas para computação não tradicional. Pretendem, em primeira análise, tentar uma aproximação do funcionamento do cérebro humano. Memória distribuída, comportamentos associativos e capacidade de cálculo distribuído são características encontradas nas redes neuronais. O aspecto não algorítmico do seu processamento tem a contrapartida da necessidade de informação previamente fornecida, à semelhança do que sucede no Homem.

Os procedimentos algorítmicos tradicionais possuem a informação armazenada com exactidão nos conjuntos de equações ou em regras bem definidas; as redes neuronais não processam usando esse tipo de meios, significando que devem ser "ensinadas" a executar determinadas tarefas. Portanto, uma fase de aprendizagem é absolutamente necessária para a rede poder processar informação. A fase de aprendizagem distingue-se, em geral, da fase de operação, onde a aprendizagem já se efectuou. Contudo, existem também alguns tipos de redes que podem aprender continuamente, mesmo durante a fase de operação.

Ensinar uma rede neuronal pode ser feito por meio de exemplos correctos para um determinado problema; a ideia é que a rede ajuste as suas ligações internas de forma a minimizar (e manter minimizado) o erro para os exemplos apresentados. Algumas requerem um outro tipo de aprendizagem: quando são usadas como memórias associativas (no espírito das *look-up tables—LUT*), é fornecido às redes um comportamento pré-determinado. Isto significa que as ligações internas são fixadas *a priori* usando os pares de entrada/saída para o problema específico — é o caso das redes de Hopfield. Em ambas as situações a informação é fornecida à rede: trata-se de aprendizagem supervisionada. Existe contudo a possibilidade de a rede fazer a aprendizagem por ela própria. Não são fornecidos exemplos, mas apenas uma regra. Na aprendizagem com reforço (*reinforcement learning*), por exemplo, todos os neurões são continuamente informados se a sua saída deve ser maior ou menor; um valor de realimentação calculado com base no ambiente (por exemplo) é usado para polarizar o comportamento da rede. Uma última técnica de aprendizagem é completamente não-supervisionada; o espaço de entrada (isto é, as suas propriedades topológicas) pode ser mapeado numa estrutura (espaço) diferente: é o caso das redes auto-organizativas.

As redes neuronais são as estruturas primordiais de processamento de informação dentro da **Neurocomputação** que, segundo Hecht-Nielsen, é a nova abordagem de processamento de informação que não requere o desenvolvimento de algoritmos ou de regras, e que muito frequentemente reduz a quantidade de programação (*software*) necessária à resolução de um problema. Formalmente, e continuando a citar Hecht-Nielsen, a neurocomputação é a disciplina tecnológica que se ocupa com sistemas de processamento de informação paralelo, distribuído e adaptativo, que desenvolve capacidade de processamento de informação em resposta à exposição a um ambiente de informação [HechtNielsen90].

As semelhanças entre as redes neuronais (artificiais) e o cérebro humano existem apenas a um nível conceptual. Actualmente, as redes neuronais não pretendem replicar o cérebro humano, apenas modelizar algumas das suas funções. As estruturas artificiais estão muito distantes das naturais: por exemplo, o número de neurões do cérebro humano vai além de 10^{11} ou 10^{12}, e o número de sinapses (ligações entre neurões) pode atingir 10^{15} a 10^{17}. Estes números são realmente elevados quando comparados com as redes artificiais que possuem actualmente apenas 10^5 a 10^6 componentes: há muitas ordens de grandeza a separar estas quantidades!

A.2. O neurão e arquitecturas gerais de redes

A.2.1. Noções básicas

Todas as redes neuronais são feitas à base de elementos designados por **neurões** ou **unidades de processamento**. Os elementos estão ligados entre si através das **ligações**, **sinapses** ou **pesos da rede**. Estes pesos contêm o "conhecimento" da rede e devem ser ajustados durante a fase de aprendizagem: o conjunto de todas as ligações da rede é por vezes designado como a **memória de longo termo** da rede. Estas ligações são ditas excitatórias ou inibitórias quando têm valores positivos ou negativos, respectivamente. Normalmente os neurões estão organizados em **camadas** ou **níveis**

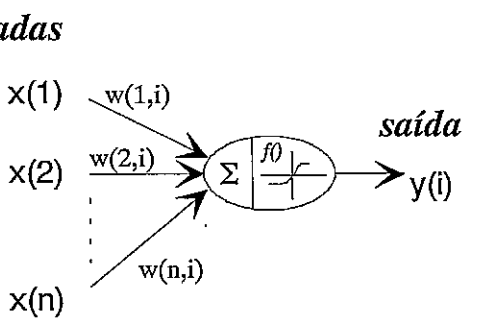


Fig. A.1 - O neurão ou unidade de processamento

A figura A.1 mostra o elemento básico com o qual todas as redes neuronais são feitas: tem várias entradas e uma saída — como as **sinapses** e o **axónio** das células biológicas. O corpo da estrutura é representado pela elipse e é onde ocorre todo o processamento

usualmente separado em duas partes: à esquerda, o estado do neurão (Σ) que é a combinação linear das entradas, e à direita, dentro da elipse, a **função de activação** ou **função de transferência** *f()*. Esta função é usada para reformatar a saída num espaço adequado: normalmente são usadas funções limitadoras não lineares. Tem-se então que a saída de um neurão é dada por:

$$y(i) = f\left(\sum_{k=1}^{N} x(k)w(k,i)\right) \qquad (A\text{-}1)$$

A função de activação é aqui representada por *f()* e *N* é o número total de entradas. De seguida ilustram-se algumas funções de activação muito usadas; o eixo horizontal refere-se ao estado do neurão e o eixo vertical a sua saída.

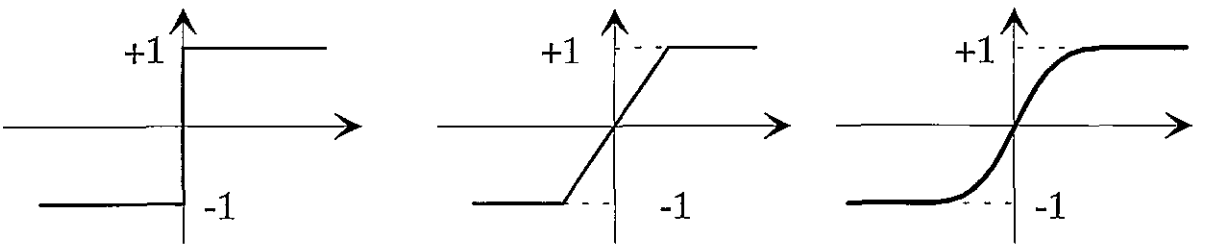


Fig. A.2 - Algumas funções de activação: limitador não linear em degrau, limitador com região linear e sigmóide.

Em certo tipo de redes a função de activação pode inclusivamente ser uma função estocástica (não determinística). A função sigmóide possui a particularidade de ser contínua bem como as suas derivadas. Este facto é importante (e até indispensável) para alguns algoritmos de aprendizagem. Nos casos representados os limites da saída situam-se no intervalo [-1,1], mas o intervalo [0,1] é também frequentemente usado. Trata-se essencialmente de conveniências de ordem computacional. Assim a função sigmóide para o intervalo [0,1] é qualquer função da categoria de funções dada por:

$$f(x) = \frac{1}{1+e^{-\beta x}} \qquad (A\text{-}2)$$

Para o intervalo [-1, 1] ter-se-ão uma categoria de funções de igual modo infinitamente deriváveis das quais faz parte, por exemplo, a tangente hiperbólica, e cuja expressão geral é:

$$f(x) = \frac{1-e^{-\beta x}}{1+e^{-\beta x}} \qquad (A\text{-}3)$$

O estado interno dos neurões pode ter valores contínuos ou discretos. Nesta última variante é frequente o uso de valores binários. No caso do estado ser contínuo, pode ser limitado ou não (por exemplo, poderá apenas assumir qualquer valor real entre 0 e +1).

A.2.2. Arquitecturas e topologias

Em teoria qualquer arranjo ou combinação de neurões ligados constitui uma rede neuronal, mas a maioria das redes usadas está organizada em camadas. Deste modo, e na maioria dos casos, os neurões dentro de uma mesma camada não têm ligações entre si. Estão ligados apenas a neurões das camadas vizinhas.

No que respeita às arquitecturas organizadas em camadas, existem vários tipos: sistemas de uma só camada ou sistemas multi-camada. A primeira camada (muitas vezes não vista como tal) é a camada de entrada. É simplesmente o ponto de entrada e não aplica qualquer processamento. A última camada, aquela de onde se lê o resultado, é a camada de saída. As restantes eventuais camadas são as chamadas camadas escondidas ou invisíveis. Contêm valores intermédios das computações do sistema e não são normalmente acessíveis, donde o seu nome de camadas escondidas.

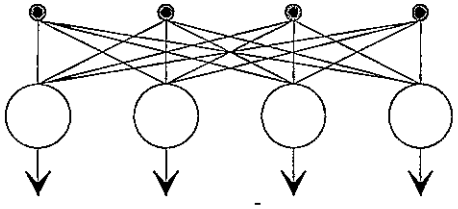


Fig. A.3 - Rede neuronal com uma só camada sem realimentação

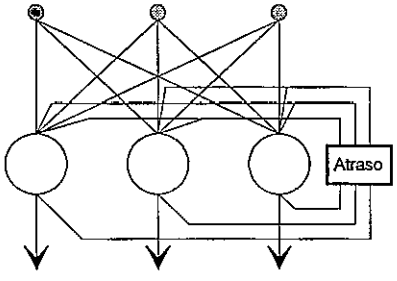


Fig. A.4 - Rede neuronal com uma só camada com realimentação.

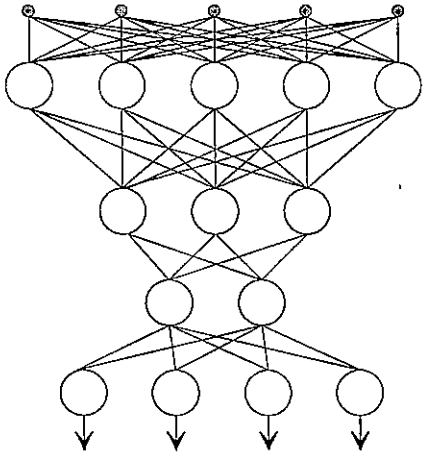


Fig. A.5 - Rede neuronal com 4 camadas (3 escondidas) sem realimentação.

As arquitecturas representadas nas figuras anteriores são exemplos de implementações para as quais existe um algoritmo específico existe. Note-se todavia que na prática raramente se usam redes tão complicadas como a da figura A.5. A maioria dos casos onde esse tipo de arquitectura é escolhido 3 camadas costumam ser suficientes e é verdadeiramente raro precisar de ir além desse número. De facto a maioria dos problemas usa apenas redes com um camada escondida, por conseguinte uma rede com duas camadas já que a camada de entrada não é contada como camada. É também de notar que se as funções de activação são lineares então não há necessidade de ter mais do que uma camada; o sistema seria sempre redutível a uma só camada — as operações matriciais que no fundo implementam as redes, como veremos, poder-se-iam reduzir à operação com uma só matriz.

Dentro de uma determinada arquitectura é possível modificar a topologia da rede e obter diferentes prestações e resultados. A questão consiste fundamentalmente em associar os neurões de uma camada em grupos pelo simples facto de eliminar algumas ligações no interior da rede.

A.3. Classificação de redes neuronais

Uma das maneiras de agrupar as redes neuronais é colocá-las em duas grandes categorias: **redes associativas** e **redes mapeadoras.** As primeiras são em geral do tipo de camada única e normalmente habilitam-se a fazer associações entre pares de vectores definidos completamente pelo utilizador. São usadas para completar ou restaurar padrões. As soluções que são supostas gerar fazem parte dum conjunto esperado e conhecido já na altura da fase de aprendizagem. Um exemplo típico é o reconhecimento de caracteres; um padrão distorcido é identificado como um dos padrões usados para o treino — a rede procura a correspondência num passo único ou iterativamente.

As redes mapeadoras, por seu lado, tem uma tarefa mais complexa. Também podem ser usadas para reconhecimento de caracteres, por exemplo, mas as suas possibilidades vão mais além, podem inclusivamente fazer previsão ou estimativas. Numa palavra, podem "criar", mais do que simplesmente fazer associações para dados ou padrões conhecidos ou esperados à partida.

A classificação de redes pode ser feita em função de outros pontos de vistas; a tabela A.1 ilustra as classificações mais comuns.

As redes auto-associativas têm uma particularidade: os valores de entrada para o treino e os padrões de saída são iguais. Ou seja, a rede é ensinada a produzir uma saída igual à entrada. Esta ideia tem aplicações específicas: podem ser usadas quando, por exemplo, se procura uma representação diferente dos dados e se tem simultaneamente uma forma de voltar à representação inicial: o caso mais comum é a compressão de imagem. As camadas escondidas, no fundo, tem uma representação diferente da entrada e da saída mas está-lhes directamente relacionada. Se a dimensão da camada escondida for menor que a entrada (e saída) e a rede aprendeu bem, então tem-se um compressor de dados!

Fluxo de dados	• Para a frente (FIR) • Realimentadas ou recorrente (IIR)	
Aprendizagem	• Não supervisionada e não competitiva • Não supervisionada e competitiva • Supervisionada	• Minimização do erro • Por reforço • Estocástica
Lei de associação	• Auto-associativa • Hetero-associativa	• Vizinho-mais-próximo • Interpoladora
Modo de aprendizagem	• Fixa • Adaptativa	

Tabela A.1 - Modos de classificação das redes neuronais.

As redes hetero-associativas são a grande maioria. As redes do tipo **vizinho-mais-próximo** fornecem saídas correspondentes a entradas o mais semelhante possível da entrada corrente, contrariamente às redes interpoladoras que podem "criar" novos padrões quando não for encontrada correspondência conhecida.

A.4. O que é usar uma rede neuronal

Os neurões são uma espécie de combinadores de sinal; adicionam as suas entradas devidamente ponderadas (com os respectivos ganhos) e modificam o resultado da soma de acordo com uma dada regra: a função de activação — este procedimento pode ser executado ou simulado em programação: os padrões podem ser vistos com vectores e o conjunto de ligações entre duas camadas como uma matriz. Se **s** é o vector de estados, **x** a entrada e **W** a matriz dos pesos, pode-se então dizer para uma camada que $\mathbf{s} = \mathbf{W} \cdot \mathbf{x}$ ou, mais explicitamente:

$$\begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_M \end{bmatrix} = \begin{bmatrix} w_{11} & \cdots & \cdots & w_{1N} \\ w_{12} & \ddots & & w_{2N} \\ \vdots & & \ddots & \vdots \\ w_{M1} & \cdots & \cdots & w_{MN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \qquad (A\text{-}4)$$

Onde N e M são as dimensões do padrão de entrada e saída respectivamente.

A expressão anterior calcula apenas o estado dos neurões de uma camada, para se obter a saída faltaria ainda aplicar a função de activação. Para múltiplas camadas dever-se-ia aplicar o processo repetidamente em cada camada. É na matriz **W** (e nas outras em caso de multi-camada) que está contido o "conhecimento" da rede. A fase de aprendizagem de uma rede neuronal consiste precisamente em determinar esta matriz. Esse processo é feito tipicamente em sucessivas iterações (redes adaptativas), ou então a matriz pode ser determinada num passo único tendo em conta os dados que se querem ensinar: trata-se das matrizes fixas, como é o caso das redes de Hopfield.

A.5. As principais abordagens da neurocomputação

O campo de aplicações de redes neuronais é muito vasto, mas pode ser sintetizado nas seguinte grandes sub-áreas: **Classificação de Padrões e Memória Associativa, Auto-organização e Extracção de Propriedades, Optimização** e **Transformações Não Lineares.** Há redes mais adequadas para algumas destas áreas e redes mais adequadas para outras [Moore92]. Diferentes algoritmos e variações nas arquitecturas podem levar à rede certa para cada problema, donde não seja correcto dizer que uma dada rede é usada para uma só aplicação específica; a memória associativa, por exemplo, pode ser obtida com muitos tipos de redes, até com as tradicionais *look-up tables*. A optimização é frequentemente feita com redes de Hopfield e suas variantes, etc.

De seguida, descrever-se-ão resumidamente alguns dos principais tipos de modelos computacionais em neurocomputação (redes neuronais) e referências aos algoritmos para os ensinar. Não esgotam de modo nenhum a variedade existente, servem apenas para ilustrar os princípios e destacar aqueles mais frequentemente usados pelos seus bons resultados. Descrições mais exaustivas e detalhadas podem ser encontradas, por exemplo, em [HechtNielsen90], [Dayhoff90], [Zurada92], [Hush93], entre muitos outros.

A.5.3. Associadores lineares

Uma correspondências entre dois espaços vectoriais pode ser vista com uma transformação matricial. Isso é o mesmo que dizer:

$$\mathbf{y}=\mathbf{W}\mathbf{x} \qquad (A\text{-}5)$$

Onde **x** e **y** são os padrões associados e **W** a matriz que traduz essa associação. Para calcular **W** todos os pares de vectores $(\mathbf{x}_k,\mathbf{y}_k)$ para $k=1,2,...,N$ tem que ser usados. Uma primeira regra, baseada na Lei da Aprendizagem Biológica de D. Hebb (1949), é calcular **W** iterativamente do seguinte modo:

$$\mathbf{W}^{new} = \mathbf{W}^{old} + \mathbf{y}_k\mathbf{x}_k^T \qquad (A\text{-}6)$$

ou, em representação não iterativa:

$$\mathbf{W} = \sum_{k=1}^{L} \mathbf{y}_k\mathbf{x}_k^T = \mathbf{Y}\mathbf{X}^T \qquad (A\text{-}7)$$

onde **X** (e semelhantemente **Y**) se define do seguinte modo:

$$\mathbf{X} = \left[\vec{\mathbf{x}}_1 \quad \cdots \quad \vec{\mathbf{x}}_L\right] = \begin{bmatrix} x_{1_1} & \cdots & x_{L_1} \\ \vdots & & \vdots \\ x_{1_M} & \cdots & x_{L_M} \end{bmatrix} \qquad (A\text{-}8)$$

M é a dimensionalidade e L o número de padrões de entrada

A expressão (A-7) é por conseguinte conhecida como **Lei de Aprendizagem de Hebb.** Designações alternativas são **Soma de Produtos Internos, Memória de Matriz de Correlação** ou ainda **Memória Associativa Linear.** A associação $\mathbf{y}_k = \mathbf{W}\mathbf{x}_k$ é exacta se todos

os $\mathbf{x}_k$ são ortogonais, o que implica que L (número de pares de vectores a associar) seja menor ou igual à dimensão de $\mathbf{x}_k$.

A.5.4. O perceptrão

O perceptrão foi introduzido por Frank Rosenblatt em 1957; o conceito original tem sido ligeiramente adaptado ao longo do tempo, mas a ideia fundamental mantém-se. É essencialmente uma rede de um ou mais unidades de processamento com saída binária. Na altura ficou muito popular pela sua capacidade de reconhecer padrões simples. Mais ainda, Rosenblatt provou que se as classes de padrões fossem linearmente separáveis então existe solução para o problema e é possível determiná-la num número finito de iterações.

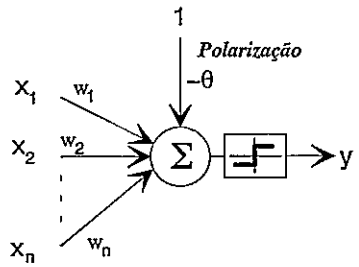


Fig. A.6 - O perceptrão

Um perceptrão elementar (um só elemento) permite a classificação de vectores em duas classes: a saída binária (+1 ou -1) decide qual a classe do vector mostrado na entrada. O parâmetro θ tem a função de "calibração de estado" e é também conhecido por polarização: simplesmente posiciona a função de activação ao longo do eixo horizontal (valores de estado do neurão). Mas esse parâmetro pode também ser visto como um outro peso da rede (designado por w_0) cuja entrada é sempre 1. A **Lei de Aprendizagem do Perceptrão** é dada por:

$$\mathbf{W}^{new} = \mathbf{W}^{old} + (y - y')\mathbf{x}_k \qquad (A\text{-}9)$$

onde y é a saída desejada e y' a saída corrente como consequência da apresentação na entrada do vector $\mathbf{x}_k$. Se (y - y') é zero não há modificações em **W**, o que significa que o seu valor satisfaz o reconhecimento de $\mathbf{x}_k$, se, ao contrário, o erro é não nulo, uma actualização (ajuste) é feita.

O espaço dos vectores de entrada é dividido em duas regiões separadas por um hiperplano; em [Lippmann87] pode-se ver uma ilustração de algumas soluções. Para o caso de se possuírem duas entradas apenas (espaço a duas dimensões) o hiperplano é uma simples recta de equação:

$$x_1 = \frac{-w_0}{w_1} x_0 + \frac{\theta}{w_1} \qquad (A\text{-}10)$$

O uso do perceptrão é limitado a problemas onde a separação linear das classes é possível. Separabilidade linear significa grosso modo que é possível estabelecer um

hiperplano no espaço de entrada de forma que os elementos de cada classe fiquem isoladas dos das outras classe. Um caso muito bem conhecido, e que se tornou no clássico que desacreditou o perceptrão, e que é muito simples de formular é o problema do **ou-exclusivo** (*XOR-problem*); é impossível traçar uma linha que separe no plano os seguintes pares de pontos em duas classes: (0, 0) e (1, 1) numa classe e (0, 1) e (1, 0) noutra classe.

A eliminação da necessidade da separação linear ocorre com o uso de redes multi-camada, como se verá mais à frente.

Em 1960 foi proposto um novo algoritmo para treinar o perceptrão. O algoritmo é designado por **Widrow-Hoff, LMS** (***least mean square***), ou ainda pelo nome de **Regra Delta**. Tem a particularidade de fornecer uma aprendizagem que minimiza o erro médio quadrático. A associação dos vectores é perfeita se as entradas forem linearmente independentes, mas mesmo as entradas não sendo linearmente independentes, a solução resultante minimiza o erro quadrático médio. É um algoritmo iterativo cuja lei é dada por:

$$\mathbf{W}(n) = \mathbf{W}(n-1) + \eta\left(\vec{t}(n) - \vec{o}(n)\right)\vec{i}^{\,T}(n) \qquad (A\text{-}11)$$

$\mathbf{W}(n) \rightarrow$ *Matriz dos pesos na iteração n*
$\vec{i}(n) \rightarrow$ *Padrão de entrada*
$\vec{t}(n) \rightarrow$ *Padrão desejado de saída*
$\vec{o}(n) \rightarrow$ *Padrão corrente de saída*
$\eta \rightarrow$ *Factor de ganho (de 0.0 a 1.0)*

É possível demonstrar que a superfície do erro é um hiperparabolóide de revolução, apresentando por conseguinte um erro mínimo. A Regra Delta é muito usada também fora dos domínios da neurocomputação.

Bernard Widrow propôs este algoritmo como o procedimento de aprendizagem das suas *ADALINES* (*ADAptive LINEar systems*) que apresentam a mesma estrutura dos perceptrões. Uma associação de várias *ADALINES* permite determinar a solução de problemas mais complexos: esses agrupamentos são designados *MADALINES* (*Many ADALINES*). *MADALINES* multi-camada podem obviar o problema da separabilidade linear. A função de activação continua sendo o limitador não linear em degrau, mas para o aprendizagem das *MADALINES* foi necessário desenvolver um novo algoritmo cujos resultados se mostraram interessantes: o MRII [Widrow88].

A.5.5. O algoritmo de retropropagação (*back-propagation*)

A introdução de redes multi-camada trouxe o problema da necessidade de regras de aprendizagem adequadas. Só em 1986 foi introduzido o algoritmo de retropropagação por intermédio de Rumelhart.

Este algoritmo permite ao sistema evoluir na direcção dos erros decrescentes, portanto em direcção a um mínimo da função de erro. O problema é que possíveis mínimos locais podem impedir o algoritmo de atingir o mínimo absoluto logo a convergência do sistema. A

figura seguinte ilustra esse risco: se o erro do sistema começar num dos pontos S, existe a possibilidade de se cair nos pontos A ou B e não mais atingir o ponto de erro mínimo M.

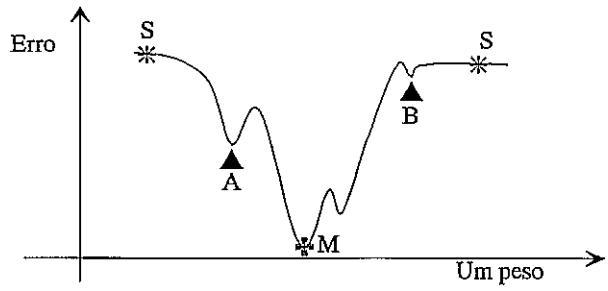


Fig. A.7 - Risco de mínimos locais no algoritmo de retropropagação.

Não se provou ainda se o algoritmo converge sempre, como foi feito para o perceptrão, mas os resultados do seu uso tem sido muito bem sucedidos em muitos problemas.

O algoritmo de retropropagação exige que o caminho dos dados seja diferenciável. É esta a razão porque a função de activação não pode ser o limitador não linear em degrau.

Tradicionalmente o erro que se procura minimizar é o erro médio quadrático, mas há outras medidas de erro que também podem ser usadas. Por exemplo, pode-se definir uma medida de erro que é a soma dos erros resultantes da apresentação de cada padrão:

$$E = \sum_{k=1}^{N} E_k$$
$$E_k = \frac{1}{2}\left\|\mathbf{d}_k - \mathbf{y}_k\right\|^2 \qquad (A\text{-}12)$$

Onde d_k é a saída desejada e y_k a saída corrente para a entrada x_k, e N é o número total de padrões

Embora o algoritmo tenha dado bons resultados, o tempo necessário para o treino é geralmente muito longo. Para acelerar esta velocidade de convergência várias técnicas foram propostas. A mais conhecida é a do termo do momento. Este termo, também designado inércia, é expresso por um parâmetro que suaviza as variações dos pesos, dando ao sistema memória sobre a aprendizagem. Outra possibilidade para acelerar a velocidade de convergência é definir factores de ganho dinâmicos: por exemplo, grande nas primeiras iterações e depois cada vez mais pequeno para afinar a convergência.

O algoritmo de retropropagação foi também generalizado para redes recorrentes, isto é redes com realimentação [Almeida87] [Almeida87b].

A.5.6. As redes de Hopfield

A rede de Hopfield é um arranjo a uma camada com realimentação (recorrente). A definição original considerava os estados dos neurões binários, e a função de activação o limitador não linear em degrau.

Esta rede é do tipo de aprendizagem fixa, o que significa que os pesos são determinados, num só passo, usando os pares de dados a ensinar à rede. Daí em diante não ocorrem quaisquer alterações nos valores dos peso da rede. A função da rede é assim em primeira amostra a de implementar *look-up tables*.

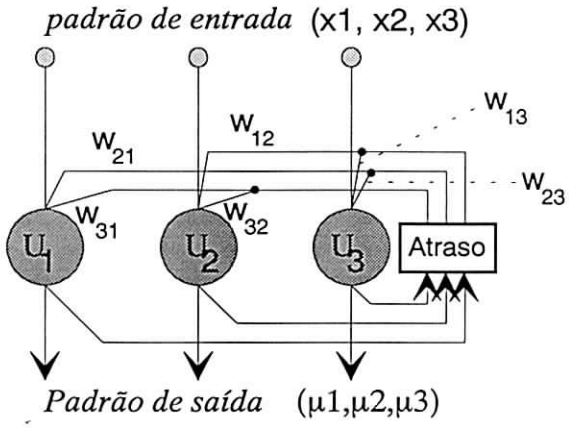


Fig. A.8 - Rede de Hopfield com entrada de dimensão 3.

Todos os neurões estão ligados a todos através de realimentação, mas matematicamente os dois pesos que ligam dois neurões entre si têm de ser iguais para haver convergência da rede. Porém, há redes que convergem mesmo sem esta condição se verificar.

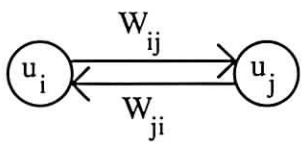


Fig. A.9 - Simetria de pesos na rede de Hopfield: $W_{ji} = W_{ij}$

Se $w_{ij}=w_{ji}$ a rede é dita simétrica, e será assimétrica caso contrário. Para a convergência, à parte a simetria, deve existir uma condição que impeça a auto-interacção, isto é, dever-se-á ter $w_{ii}=0$.

Estudos experimentais mostram que o número máximo de padrões armazenados não deve ser maior do que 0.15N, onde N é o número de unidade de processamento.

Em 1984 o próprio Hopfield introduziu a versão contínua da rede que toma o seu nome. Os padrões e o saídas dos neurões são agora números reais (intervalo [0,+1]), e a função de activação passou a ser a sigmóide.

As redes de Hopfield são também muito usadas em problemas de optimização. O exemplo clássico é o problema do vendedor viajante (*Travelling Salesman Problem*) que tem de percorrer várias cidades optimizando os seus recursos. Em [Dayhoff90] e [Zurada92] encontram-se mais detalhes sobre o uso de redes Hopfield para problemas de optimização.

A.5.7. Competição e auto-organização

Nesta categoria de redes neuronais é introduzida uma nova forma de aprendizagem. Não se fornecem exemplos como no caso de retropropagação, e também não se fornece uma matriz de peso pré-calculada como no caso de Hopfield. Trata-se aqui de aprendizagem não supervisionada. A rede (isto é, o algoritmo) ajusta os pesos internos de forma a que futuras entradas sejam representadas pela sua classe: o **centróide** de um grupo de vectores vizinhos topologicamente. A classificação não supervisionada é também conhecida por **aglomeração** — formação de aglomerados (*clustering*).

Surge aqui uma nova ideia: a existência de competição, que significa que vai ocorrer um tipo de selecção A estrutura central destas redes é uma camada competitiva de neurões completamente ligados às entradas. As unidades de saída terão valores binários, mas só uma delas (a do elemento vencedor) será uma saída de valor alto, e só esse neurão actualizará os seus pesos. É a chamada aprendizagem competitiva ou de Kohonen, sendo por vezes chamada de *winner-takes-all* ("**o-vencedor-ganha-tudo**"). A aprendizagem competitiva necessita de muitos dados de entrada para efectuar uma treino.

Quando a rede é definida, por exemplo, com 2 saídas apenas (esperando-se portanto apenas duas classes), os dados apresentados vão gradualmente modificando os pesos numa forma tal, que na fase de operação, o hiperespaço das entradas vai ser dividido em duas partes. É todavia importante realçar que a distribuição dos vectores de entrada durante a fase de treino é crucial: se os dados são apresentados seguindo uma distribuição não uniforme, então uma separação não uniforme ocorrerá quando novos dos forem apresentados para classificação.

A aprendizagem é feita nas seguinte fases:

1. Calcular as distâncias (normalmente Euclidianas) do vector de entrada a cada vector de pesos associado a cada unidade. O neurão "vencedor" é aquele que tiver a menor distância *d*:

$$d_j = \sum_{j=1}^{N} \left(x_i(t) - w_{ij}(t)\right)^2 \qquad (A\text{-}13)$$

2. Actualizar os pesos (apenas os do "vencedor", os restante não se alteram):

$$\mathbf{w}_i^{new} = \mathbf{w}_i^{old} + \alpha\left(\mathbf{x} - \mathbf{w}_i^{old}\right) \qquad (A\text{-}14)$$

Com $0 < \alpha < 1$; um valor típico é 0.7

No mundo natural, tais aprendizagens competitivas têm também sido encontradas. Todavia, a política de encontrar o elemento "vencedor" é comandada por aquilo a que se chama **inibição lateral**. Nesse caso os neurões estão ligados lateralmente entre si (dentro da mesma camada) e mandam sinais inibitórios a todos os seus vizinhos. A intensidade deste sinal, em cada instante, é proporcional ao estado do neurão, logo o "vencedor" envia o sinal

inibitório mais intenso. Mais ainda, cada unidade envia sinais excitatórios a si própria. O raio de acção destes sinais varia com a distância geométrica dos neurões. A intensidade da interacção com a distância é descrita por uma curva designada por **Chapéu Mexicano** (*Mexican Hat*) cuja ilustração básica se dá abaixo.

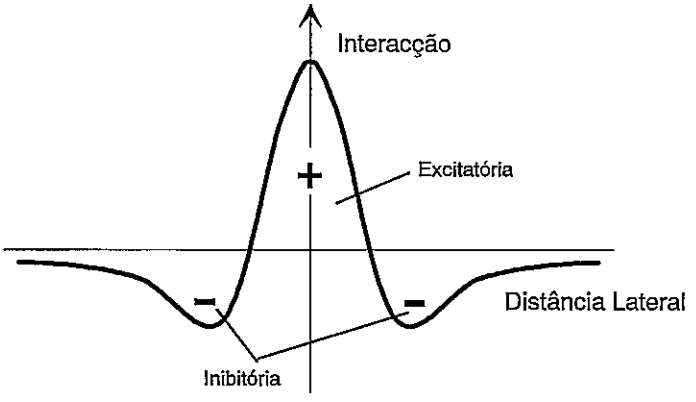


Fig. A.10 - "Chapéu Mexicano": a interacção de neurões com a distância na aprendizagem competitiva e inibição lateral.

A.5.7.1. Mapas auto-organizativos

Neste espírito de competição e inibição lateral, Kohonen propôs aquilo a que chamou uma arquitectura de Mapa Auto-organizativos. É fundamentalmente uma estrutura competitiva como descrito anteriormente, mas a actualização dos pesos não afecta apenas o neurão "vencedor" mas também uma sua vizinhança que varia dinamicamente: no início é grande e depois diminui lentamente durante o tempo de treino, à medida que mais entradas são fornecidas. Esta camada competitiva pode ter uma arranjo geométrico tal se podem definir vários tipos de vizinhanças (fig. A.11).

O sistema (a rede e o algoritmo de aprendizagem) proporciona organização (classificação) baseado exclusivamente nas entradas. Esta organizações foram matematicamente demonstradas como sendo possíveis para vectores unidimensionais, isto é, simples valores numéricos [Kohonen89].

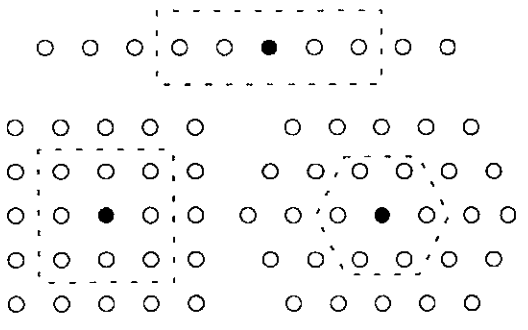


Fig. A.11 - Exemplos de tipos de vizinhanças nos mapas auto-organizativos (linear e bidimensional)

A.6. O Teorema de Kolmogorov

Para finalizar esta pequena descrição dos princípios das redes neuronais enumera-se um teorema muito curioso que se relaciona com as dimensões de uma rede neuronal para a resolução de problemas de mapeamento (correspondências entre espaços vectoriais).

Teorema da existência de uma Rede Neuronal de Mapeamento de Kolmogorov:

> *"Dada uma função contínua f:*$[0, 1]^n \rightarrow \mathbf{R}^m$, $f(x)= \mathbf{y}$, *f pode ser implementada exactamente por uma rede neuronal não recorrente com 3 camadas (uma camada escondida) com* ***n*** *entradas,* ***(2n+1)*** *unidades de processamento na camada escondida e* ***m*** *unidades de processamento na camada de saída."*

As unidades da camada escondida terão as saídas dadas por:

$$z_k = \sum_{j=1}^{n} \lambda^k \psi\left(x_j + k\varepsilon\right) + k \qquad (A\text{-}15)$$

onde λ é uma constante real e ψ é uma função real monótona crescente; ambas são independentes de *f*, mas não de *n*. A constante ε é um número racional $0 < \varepsilon \leq \delta$, onde δ é uma constante positiva arbitrariamente escolhida. As unidades de saída têm a seguinte função de activação:

$$y_i = \sum_{k}^{2n+1} g_i\left(z_k\right) \qquad (A\text{-}16)$$

onde as funções g_i (i=1,2,...,m) são reais e contínuas (e dependem de *f* e ε).

Não são dados exemplos para os g_i nem para ψ, nem é dado qualquer valor para ε. Este teorema determina a existência de uma rede neuronal para permitir a correspondência desejada mas não dá à partida formas de determinar os pesos da rede. Esses terão que ser determinados com um algoritmo como o de retropropagação, por exemplo.

Apêndice B - Descrição das infra-estruturas e ambiente de trabalho

A grande maioria do trabalho desenvolvido foi feito no âmbito de um projecto de investigação coordenado pelo Prof. Dr. João G. M. Gonçalves, no Centro Comum de Investigação da Comissão Europeia em Ispra, Itália. Trata-se de um projecto cujos objectivos incluem o estudo e desenvolvimento de um protótipo de um sistema robótico para proceder à verificação (e eventual manipulação) de componentes numa zona de armazenamento de produtos radioactivos [Gonçalves91] [Gonçalves93] [Gonçalves93b].

No momento em que o trabalho descrito ao longo deste texto foi iniciado, o sistema consistia já num conjunto de ferramentas e acessórios que se descreverá de seguida.

B.1. Enquadramento no projecto global

Como se pode verificar, por exemplo, em [Gonçalves93], o projecto global assume a existência de três fases em direcção à autonomia, das quais só a fase I se considerava, nessa altura, atingida. Era portanto imperativo prosseguir em direcção às fases seguintes de autonomia. Esta era uma razão institucional. Na verdade, as preocupações na altura giravam em torno do problema do desvio de obstáculos (essencialmente os desconhecidos) e da recuperação de trajectórias. O sistema dispunha já de um método de localização com algumas limitações de aplicabilidade, mas bastante preciso [Sequeira93b], e possuía também um método razoável para o traçado e execução (em malha aberta, porém) de uma trajectória **clotoidal**.

O problema por resolver era formulado mais ou menos do seguinte modo: a localização seria efectuada a intervalos de tempo mais ou menos regulares (alguns minutos, por exemplo), por isso não seria de contar com ela para a execução segura de uma trajectória; essa execução de trajectória é que deveria ser vigiada pelo sistema de ultra-sons para evitar embates com obstáculos desconhecidos no momento do traçado. O sistema ultra-sónico poderia ainda fazer a recuperação da trajectória perdida temporariamente. Esta visão não é propriamente de autonomia completa, mas poderia ser um passo para ela.

Outra questão que se esperaria para um sistema à base de ultra-sons era o de ser capaz de atravessar uma porta ou contornar um canto de parede automaticamente, sem intervenção humana. Aqui surgem mais algumas pretensões de autonomia.

À parte esta dupla missão dum sistema baseado em ultra-sons (desvio de obstáculo e recuperação de trajectória, e navegação automática em certas circunstâncias) esperava-se que os ultra-sons, e isso estava já implícito nas funções anteriores, fossem utilizados como sistema de segurança e servir como entrada de um processo de alta prioridade (*watchdog*) que parasse o veículo em caso de eminência de colisão. Esta componente de segurança, se bem que, pelo menos em teoria, bastante mais simples que as outras tarefas, era muito importante.

B.2. O sistema previamente existente

O equipamento relacionado com esta aplicação consistia essencialmente numa plataforma móvel do tipo *Robuter* e em *Workstations* do tipo *Sun Sparcstation* para servir de consola para o utilizador. O sistema era fundamentalmente orientado para comando do robot por parte de um operador através de uma consola, estando portanto a componente de navegação automática em fase pouco desenvolvida, mas estando a componente da *interface* Homem-máquina num estado já assente e evoluído [Sequeira92] [Silva92].

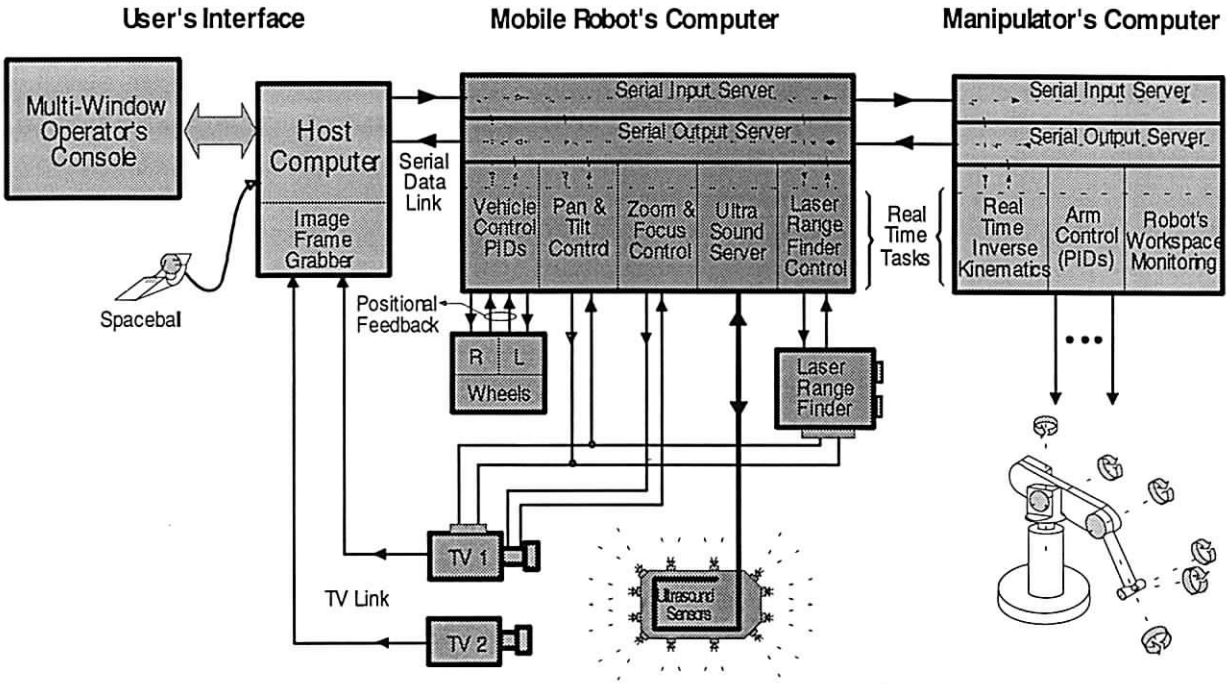


Fig. B.1 - Arquitectura *hardware* do sistema completo inicial (adaptado de [Sequeira93])

Os ambientes de programação consistiam em UNIX do lado da estação de trabalho, e o sistema de tempo real ***Albatros*** no robot. A *interface* gráfica para o controlo do robot é desenvolvida em X11R5, OSF/Motif e PHIGS.

Uma descrição panorâmica e mais completa do sistema nesta fase pode ser encontrado em [Gonçalves93] e [Gonçalves93b], mas fica de seguida uma breve enumeração das arquitecturas de *software* e principais ferramentas de *interface* desenvolvidas.

B.2.1. As arquitecturas de *software* no robot e na estação de controlo

As arquitecturas de *software* assentavam no princípio de existência de um servidor, que só ele acedia à linha de comunicação entre robot e a estação de controlo, e de vários clientes (processos) que comunicavam com esse servidor a pedir ou enviar informação de um lado para o outro — tratava-se do **Servidor de Comunicações**. Esta designação sugere uma orientação dos métodos e processos para as comunicações, o verdadeiro apanágio do sistema

dado assentar num recurso muito escasso e muito requisitado: a linha de comunicação série via rádio-modem a 9600 *baud full duplex.*

A arquitectura ilustrada na figura B.2 é uma representação da arquitectura global de *software* tal como era vista antes deste trabalho em navegação local. Alguns módulos representados não estavem desenvolvidos na altura, e a sua implementação final acabou por ser ligeiramente diferente, nomeadamente no caso do robot. Ao nível da estação de controlo, a arquitectura de *software* não precisou de mudar em quase nada.

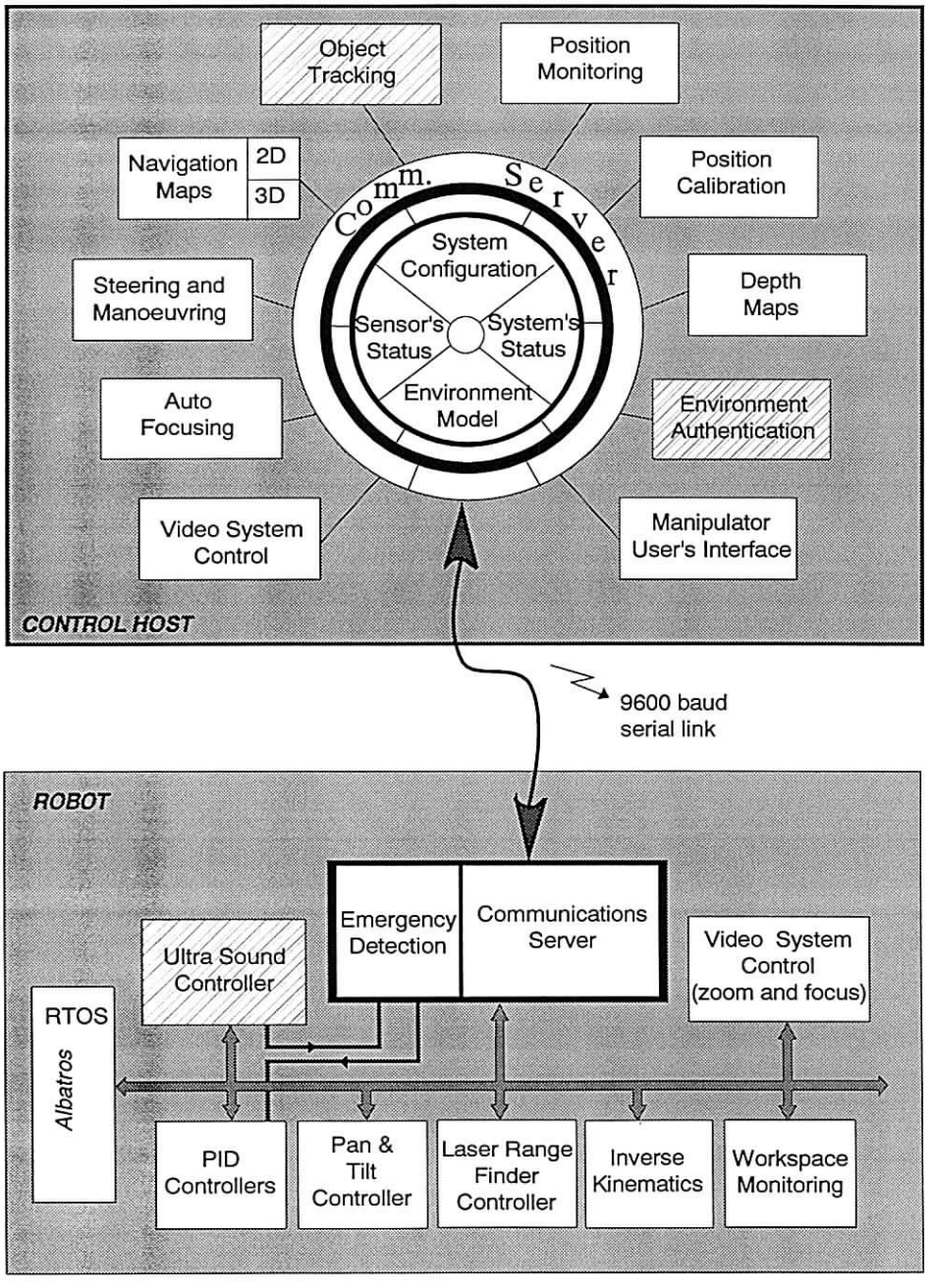


Fig. B.2 - Arquitectura global de *software* do sistema. A sombreado indicam-se módulos ainda por desenvolver nos primeiros tempos. (adaptado de [Sequeira93])

Assim, o *software* no robot, depois de implementada a facilidade de navegação local, ficou ligeiramente diferente. Aquilo a que se chamava detecção de emergências era um método que verificava o corte de comunicações e parava o robot nessas circunstâncias. Foi introduzida uma verdadeira detecção de iminência de colisão, havendo portanto uma certa dissociação entre o **servidor de comunicações** e todas as funções de **detecção de emergências,** ficando-lhe apenas associado a detecção de falhas de comunicações que pode inclusivamente ser ignorada para permitir verdadeira autonomia. A figura B.3 ilustra as modificações introduzidas:

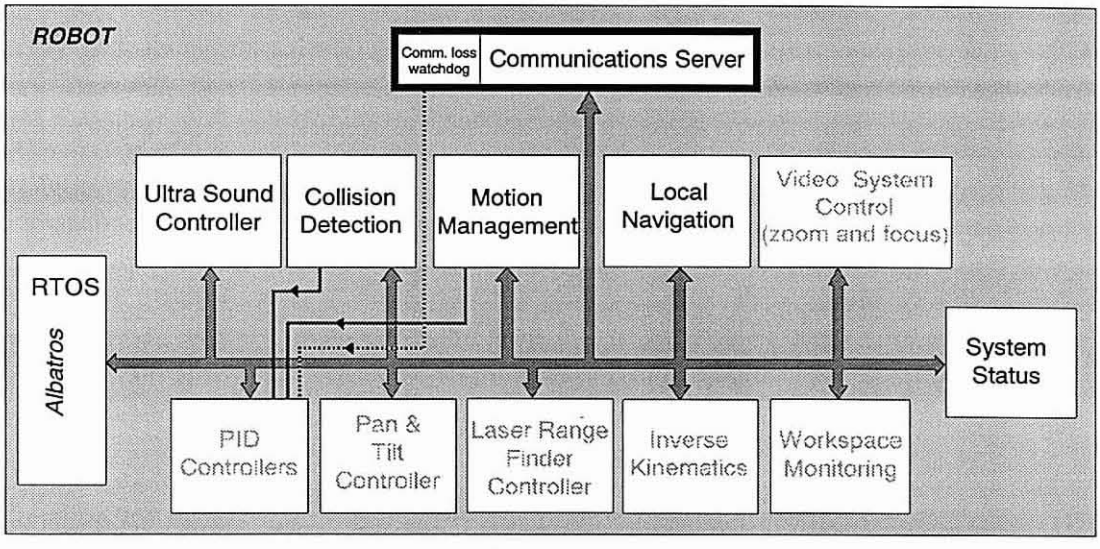


Fig. B.3 - Arquitectura de software após as modificações. Repare-se na representação explícita do estado do sistema.

B.2.2. Os programas de controlo e acesso aos equipamentos

Na estação de controlo, o cerne da *interface* gráfica com o utilizador, existiam algumas ferramentas fundamentais que permitiam:

- controlar manualmente o robot em velocidade (**xguide**);
- uma representação gráfica do robot no ambiente usando os dados de odometria fornecidos continuamente pelo robot via linha série e o modelo da sala a duas dimensões (**xroom**) ou a três dimensões (**xroom3d**);
- uma representação gráfica dos valores sensoriais instantâneos dos ultra-sons fornecidos continuamente pelo robot usando igualmente a linha série (**xusound**);
- definir trajectórias **clotoidais** a ser executadas em malha aberta e com controlo por posição através de especificação interactiva dos pontos de passagem por parte do utilizador(**xtraj**).

Além destes módulos gráficos existiam ainda outros, mas possuíam uma importância secundária para a navegação em si, como é o exemplo do controlo do braço (**xarm**) ou das

câmaras de vídeo e lentes (**xcamera**). A descrição da maioria destes módulos pode ser vista com detalhe em [Sequeira93].

B.3. Equipamento e programas adicionados

Para o desenvolvimento do trabalho descrito ao longo deste documento foram a seu tempo acrescentados componentes e elementos quer de *hardware* quer ao nível de módulos na *interface* gráfica. De seguida descrevem-se essas alterações.

B.3.3. Sistema VXI/VMEbus-Sbus

Para obtenção de maior velocidade na transferência de programas para o robot, foi instalado um sistema da *National Instruments* [NationalInst92] que permite a transferência rápida entre as memórias centrais de um sistema à base de **SBus** (*Sun*) e outro à base de **VME/VXIbus** (VME no robot). As repercussões no tempo de desenvolvimento dos programas para o robot foram extraordinariamente melhoradas, uma vez que se passou de tempos de transferência de código de muitos minutos (5-10) para tempos de poucos segundos ou bastante menos ainda. O código compilado a executar no robot já não precisava de ser convertido em **S-records** e depois transferido via linha série da estação de trabalho onde tinha sido compilado com um *cross-compiler* adequado. Foram pois escritos programas que colocavam directamente esse código compilado nos endereços apropriados da memória do robot de forma transparente através deste sistema Sbus-VXI/VME.

B.3.4. Interface gráfica - paineis de controlo extra

Logo no princípio surgiu a necessidade de ter uma representação numérica bem visível das coordenadas do robot no ambiente. Esse facto era importante particularmente durante as sessões de aquisição de dados. Para o efeito foi escrito código para um painel desse tipo que se encaixava bem no ambiente gráfico onde se inseria, e dialogava com os módulos da arquitectura de *software* tal como todos os restantes. Foi designado por **xshowpos** e abaixo ilustra-se o aspecto do painel.

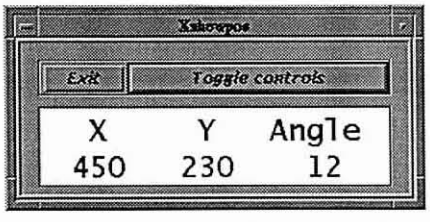


Fig. B.4 - O painel com a posição instantânea fornecida pelo próprio robot.

Para o estudo das estratégias de disparo de sensores de ultra-som era necessário um controlo directo e fácil sobre a activação de todos e cada sensor de ultra-som, bem como a selecção de agrupamentos ou tempos de atraso especiais. Para permitir essa tarefa foi escrita

uma ferramenta gráfica apropriada e designada por **xsensors.** É um quadro muito dinâmico e adaptável que possuía três modos de apresentação conforme as necessidade de controlo.

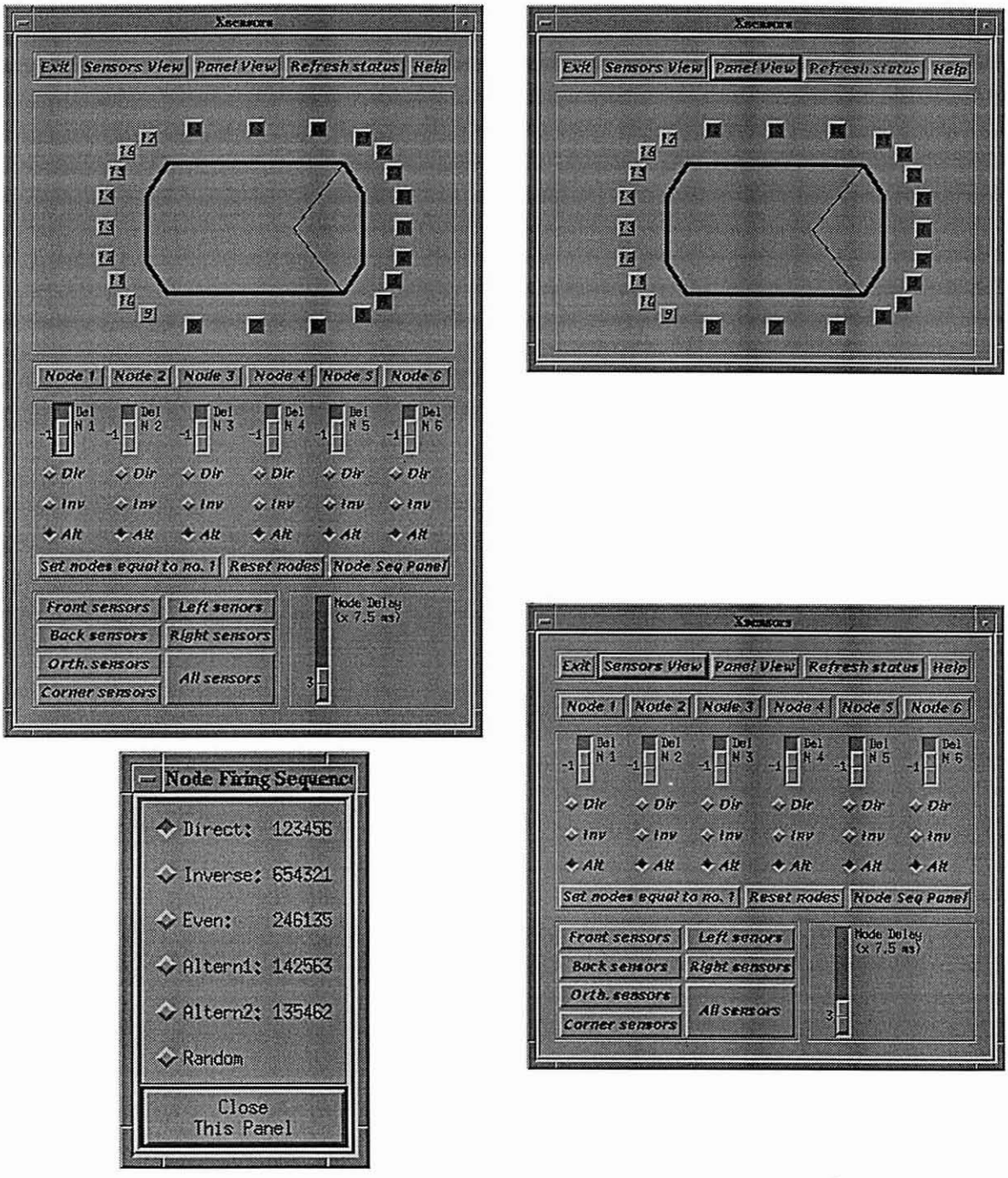


Fig. B.5 - O painel para a activação e selecção dos sensores em grupos ou individualmente. À direita duas variantes simplificadas. À esquerda, em baixo, o subpainel para a selecção da ordem de disparo dos nodos.

Um outro painel introduzido nas fases finais do trabalho tem a ver com o controle da navegação autónoma e alguns parâmetros para o detector de emergências. Trata-se de **xnav**.

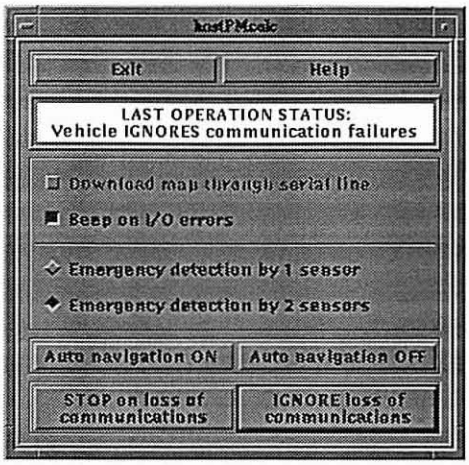


Fig. B.6 - Painel de ajuste dos parâmetros relacionados com a navegação autónoma.

Um último elemento de *interface* gráfica é o programa **xgrid**. Na verdade este foi o programa principal para o estudo e visualização das grelhas, comportamento das redes neuronais, aplicação de dados em tempo real vindos do robot, comparação dos diferentes métodos para gerar os mapas, simulação de dados e navegação com geração de sequências animadas, geração dos conjuntos de treino em modo não gráfico, produção de saída gráfica em formato PostScript, estabelecimento de comunicação e troca de informação com outros processos UNIX, etc, etc, etc. É um programa de grande envergadura permitindo largas dezenas de opções na linha de comando e que evoluiu ao longo de mais de 20 versões durante 18 meses.

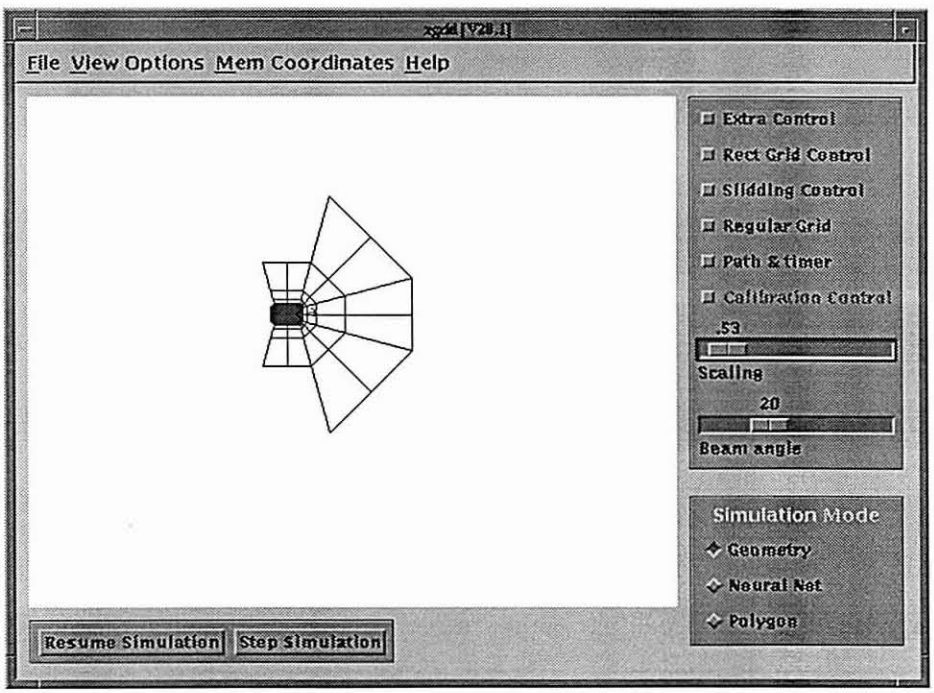


Fig. B.7 - Quadro principal do programa xgrid.

A aplicação possuía diversos paineis secundários para algumas facilidades particulares, e que se passa a listar de seguida:

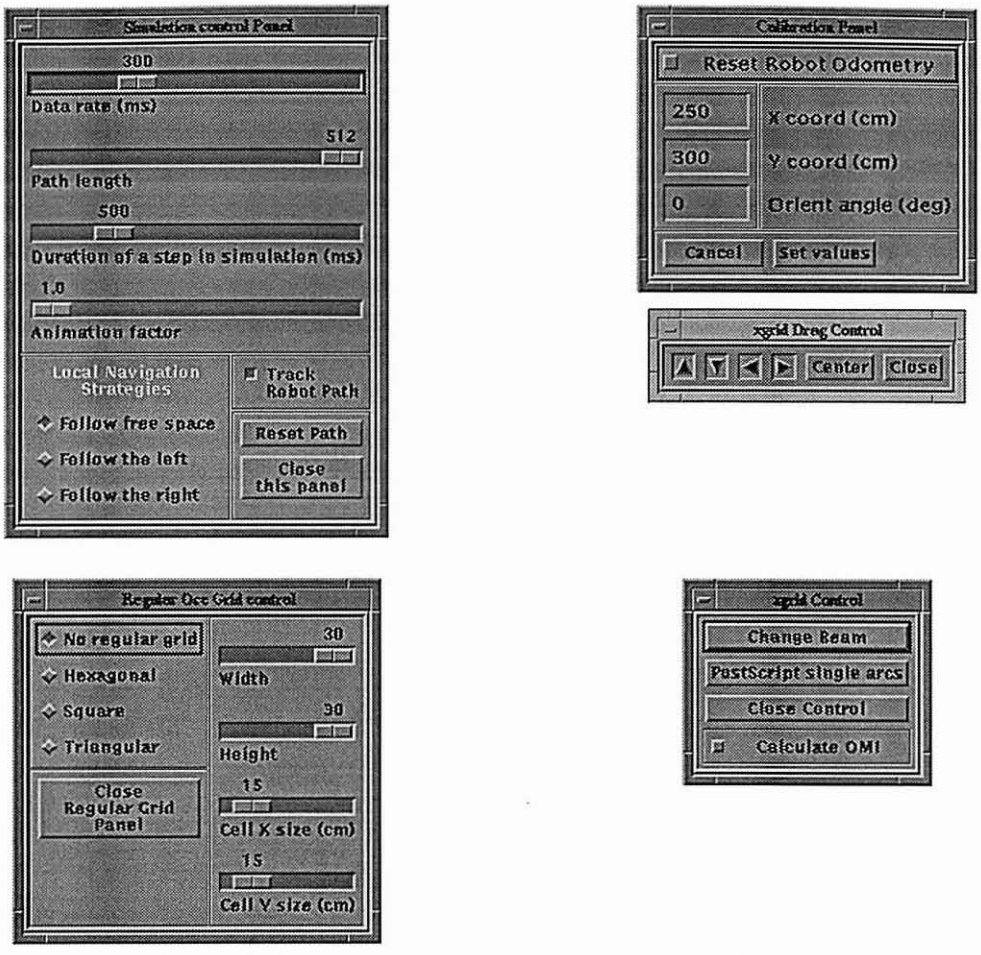


Fig. B.8 - Paineis auxiliares de controlo na aplicação xgrid.

B.3.5. Adicionais à arquitectura de *software* e *hardware* do robot

Para permitir a introdução de componente autónoma na navegação do robot foi necessário adaptar ligeiramente a arquitectura de software no robot. Na verdade as modificações foram mínimas no sentido em que a arquitectura existente era adequada e funcionante. Simplesmente tornava-se necessário introduzir a possibilidade de outros agentes poderem comandar o robot: a detecção de emergências deveria poder pará-lo e a navegação local seria capaz de o comandar quando não existisse controlo manual, que era o único previsto.

Este processo passou pela inserção dos processos adequados no sistema operativo de tempo real, e as tais ligeiras adaptações do servidor de comunicações de forma a que permitisse o funcionamento de outros módulos mesmo na ausência de qualquer contacto com a estação de controlo. Em caso de perda de comunicações, o servidor de comunicações pura e simplesmente o parava robot. Após as modificações introduzidas, e na situação de perda de comunicações, as acções a executar pelo robot são inclusivamente parametrizáveis remotamente enquanto há comunicações. Deste modo o servidor de comunicações permite a

outros processo mais habilitados a tomada de decisões quanto ao movimento a efectuar ou não.

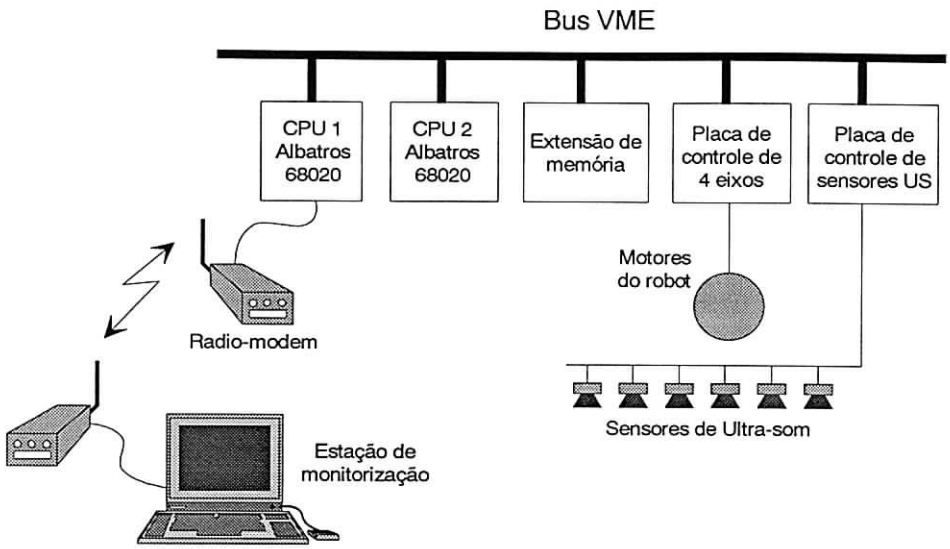


Fig. B.9 - Esquema das principais unidades *hardware* incluindo o segundo CPU instalado e dedicado ao cálculo das redes neuronais.

Do ponto de vista de *hardware*, as modificações introduzidas no robot resumem-se à adição de mais uma placa de CPU devido à falta de recursos computacionais (fig. B.9). Introduziu-se assim o multiprocessamento com particular destaque para os problemas de comunicação entre processos em processadores e espaços de memória distintos. A figura B.10 ilustra os principais processos de tempo real em cada CPU, bem como os canais de comunicação entre eles.

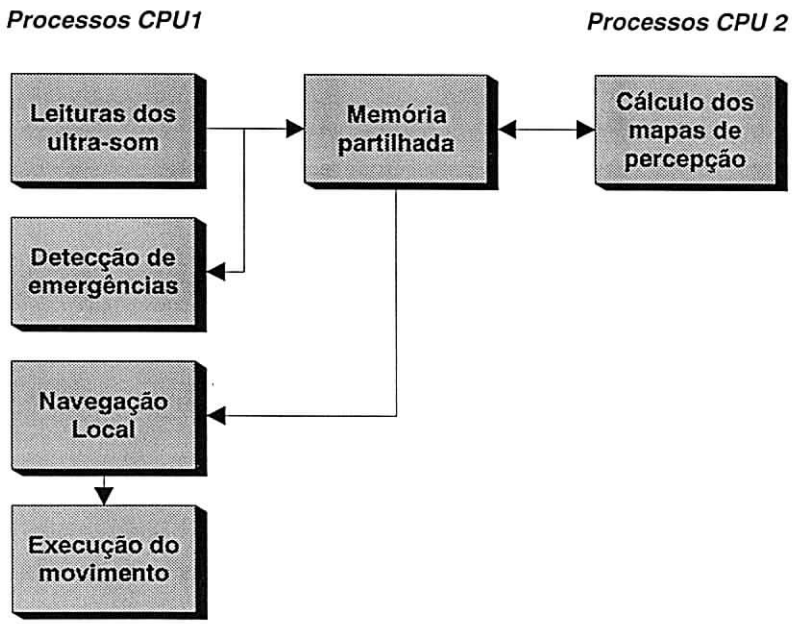


Fig. B.10 - Principais processos de tempo real nos dois CPUs. A memória partilhada é o meio de comunicação entre os processos em processadores diferentes.

Estes processos são tarefas de um sistema operativo de tempo real, e como tal possuem, entre outras as seguintes propriedades:

- executam independentemente e de uma forma contínua.
- têm diferentes prioridades consoante a sua importância.
- comunicam através de memória partilhada (os processos no mesmo CPU usam a memória local).

Apêndice C - Variante do algoritmo para o teste de inclusão poligonal

Um problema frequentemente abordado na geometria computacional é o caso da determinação da condição de inclusão de um ponto numa região do plano.

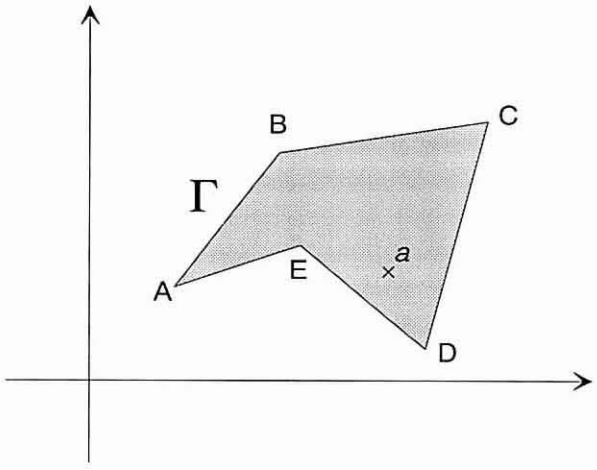


Fig. C.1 - Ilustração de uma condição de pertença de um ponto a um polígono.

O algoritmo tradicional baseia-se no conceito de semi-rectas com origem no ponto em causa, e no número de intersecções que estas fazem com o polígono que delimita a região a testar [Preparata85]. Carece ainda o algoritmo de um teste subsequente para o caso da intersecção coincidir com um ou mais vértices do polígono. Nesse caso, dever-se-ia repetir esta série de passos após uma rotação infinitesimal das semi-rectas até a condição de intersecção de um vértice do polígono deixar de existir.

Este algoritmo, se bem que simples e de baixo custo de computação *{O(n)}*, sofre da deselegância da necessidade da iteratividade. Todavia, a variante proposta em [Sedgewick84] resolve a questão de uma forma diferente e mais funcional.

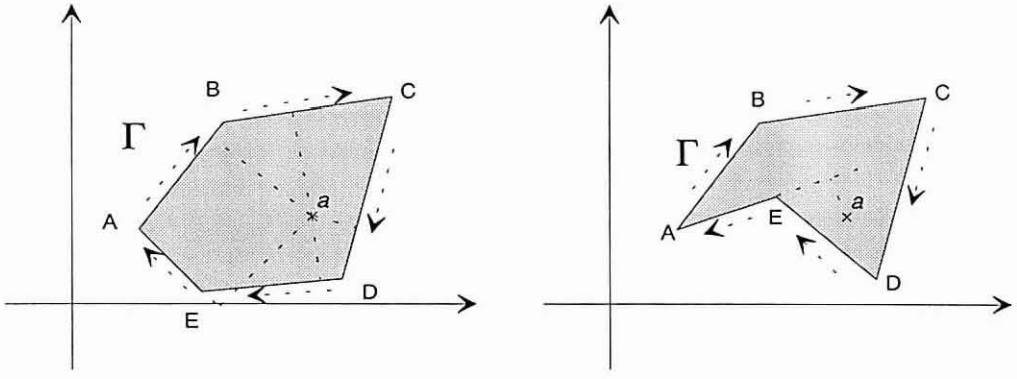


Fig. C.2 - O método da circulação (constância da posição relativa do ponto de teste a todos os segmentos do polígono) falha em polígonos côncavos: à direita, o segmento EA "vê" o ponto *a* à sua esquerda contrariamente a todos os segmentos restantes que o "vêem" à sua respectiva direita.

Uma primeira alternativa para este algoritmo da inclusão de um ponto por um polígono foi procurar uma solução com base na definição de um sentido para a linha poligonal que delimita a região do plano. Em seguida, definir-se-á uma série de ângulos orientados formados

por cada segmento de recta da linha poligonal e pelo segmento definido pela origem do segmento da linha poligonal e pelo ponto em questão. O ponto pertencerá à região poligonal se todos os ângulos orientados definidos possuírem o mesmo sentido (ângulos nulos implicam que o ponto se encontra na fronteira da região poligonal). Isto é equivalente a dizer que todos os segmentos do polígono (depois de convencionado um sentido) "vêem" o ponto em causa sempre à sua esquerda ou à sua direita. Contudo, este algoritmo exige que o polígono seja convexo, situação dominante nos problemas tratados neste trabalho, mas não a única, por isso remetido para segundo plano.

A segunda alternativa é uma proposta baseada em elementos da análise complexa, mais particularmente no *Teorema dos Resíduos* (ver por exemplo [Wylie82]) que afirma:

> Se Γ é uma curva fechada (contorno) do plano, e *f(z)* é uma função analítica no domínio definido por Γ inclusive, excepto num número finito de pontos do interior de Γ, então:
>
> $$\oint_{\Gamma} f(z)dz = 2\pi j \sum_{k=0}^{n} r_k \qquad (C\text{-}1)$$
>
> onde r_K são os *n* resíduos de *f(z)* nos pontos singulares no interior de Γ.

O valor do *resíduo* de uma função *f(z)* num dado ponto *a*, polo de multiplicidade *m* dessa mesma função, dentro de um certo contorno Γ do plano complexo, é dado pelo coeficiente de ordem -1 (α_{-1}) da expansão de Laurent de *f(z)*, mas pode ser igualmente calculado graças a um outro teorema do ***Cálculo dos Resíduos*** por:

$$\alpha_{-1} = \frac{1}{(m-1)!} \lim_{z \to a} \frac{d^{m-1}}{dz^{m-1}} \left[(z-a)^m f(z) \right] \qquad (C\text{-}2)$$

que, para o caso de um polo de ordem 1 resulta em:

$$\alpha_{-1} = \lim_{z \to a} \left[(z-a) f(z) \right] \qquad (C\text{-}3)$$

Recorde-se a propósito o Teorema de Cauchy (ou Cauchy-Goursat† numa forma mais geral) que afirma:

> Se R é uma região, simplesmente ou multiplamente ligada, cuja fronteira Γ é seccionalmente linear, e se *f(z)* é analítica e *f'(z)* é contínua dentro de R e na sua fronteira Γ então:
>
> $$\oint_{\Gamma} f(z)dz = 0 \qquad (C\text{-}4)$$

† Edouard Goursat (1858-1936) demonstrou este teorema sem impor a hipótese que *f'(z)* é continua!

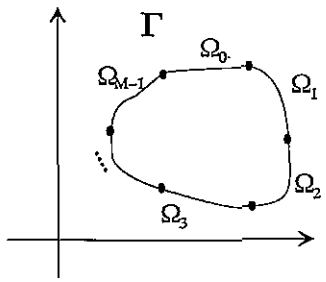


Fig. C.3 - Fragmentação arbitrária de um contorno fechado em linhas.

Se o contorno Γ for decomposto em M partes Ω_i com $0 \leq i < M$, como ilustrado na figura anterior, pela definição de integral de contorno como sendo um integral de linha onde a trajectória de integração é uma curva fechada, tem-se que:

$$\oint_{\Gamma} f(z)dz = \int_{\Omega_o} f(z)dz + \int_{\Omega_1} f(z)dz + \cdots + \int_{\Omega_{M-1}} f(z)dz \qquad (C\text{-}5)$$

Deste modo, se definirmos a função $f(z)$ do seguinte modo,

$$f(z) = \frac{1}{z-a} \qquad (C\text{-}6)$$

vem que,

$$\int f(z)dz = P_f(z) = \ln(z-a) + C \qquad (C\text{-}7)$$

e, como a é um polo de ordem 1 da função $f(z)$, aplicando a expressão (C-3), tem-se que:

$$\alpha_{-1} = \lim_{z \to a}\left[(z-a)\frac{1}{(z-a)}\right] = 1 \qquad (C\text{-}8)$$

Se Γ for uma linha poligonal constituída por simples segmentos de recta $[A_iB_i]$, e sabendo que:

$$\int_{z_0}^{z_1} f(z)dz = P_f(z_1) - P_f(z_0), \qquad (C\text{-}9)$$

onde z_0 e z_1 são os extremos de um trajecto qualquer onde $f(z)$ é analítica, recorrendo à expressão (C-5) virá que:

$$\oint_{\Gamma} f(z)dz = \ln(z-a)\Big|_{A_0}^{B_0} + \ln(z-a)\Big|_{A_1}^{B_1} + \cdots + \ln(z-a)\Big|_{A_{M-1}}^{B_{M-1}} = \sum_{i=0}^{M-1} \ln\left(\frac{B_i - a}{A_i - a}\right) \qquad (C\text{-}10)$$

Usando o próprio teorema dos resíduos (C-1), e sabendo ainda que a função só tem um resíduo, e de valor unitário, obter-se-á finalmente:

$$\sum_{i=0}^{M-1} \ln\left(\frac{B_i - a}{A_i - a}\right) = 2\pi j \qquad (C\text{-}11)$$

Esta condição (C-11) é portanto consequência do ponto a ser um polo da função $f(z)$ na região limitada pelos segmentos de recta, tal como enunciado. Isto é, se o ponto a pertence a

essa região então a condição verifica-se. Se a condição não se verificar, então o ponto *a* não pertence à região, logo não está incluído no polígono.

Note-se que se a condição não se verificar para a função *f(z)* definida, então o teorema de Cauchy (C-4) garante que o primeiro membro da expressão (C-11) terá valor nulo.

O algoritmo consiste então em calcular a expressão (C-11), e concluir por simples comparação (com as devidas precauções inerentes à precisão da aritmética em vírgula flutuante) se o ponto *a* pertence ou não ao domínio Γ.

Pelas condições de validade da expressão (C-9) este algoritmo pode não ser legítimo quando o ponto *a* cai sobre o contorno, em particular sobre um dos extremos dos vários segmentos, situação que deve ser testada antes de levar a cabo o algoritmo em si para evitar anomalias na computação numérica.

Este algoritmo é computacionalmente mais intenso que o tradicional [Preparata85] apenas pelo facto de recorrer ao cálculo de logaritmos complexos, mas é igualmente do tipo *O(n)*, isto é, de custo computacional proporcional ao número de lados do polígono. A sua vantagem está na necessidade da passagem única no cálculo (algoritmo não iterativo), e na sua elevada elegância, quando se dispõe, é claro, de funções numéricas para o cálculo complexo. Para efeitos computacionais a expressão (C-11) pode ser expandida e assim eliminar as operações de divisão de números complexos, mas em contrapartida quase duplicará o número de logaritmos de números complexos!

Apêndice D - Elementos para um simulador de ultra-sons

Um simulador completo de sensores de ultra-som é necessariamente muito complexo pela própria natureza do princípio físico e pelo complexo modelo de radiação (ainda assim aproximado) do transdutor piezoeléctrico. Outro elemento de complexidade a adicionar é a dependência com o ambiente de medição: ângulos de incidência, extensão da área reflectora e textura dos corpos reflectores são exemplos disso mesmo.

Todavia, uma série de simplificações permite prosseguir com a tarefa de simulação. A primeira simplificação foi a de considerar que o feixe de ultra-sons é perfeitamente cónico e os lóbulos secundários de radiação são desprezados. Segundo, a amplitude do feixe foi fixada num valor que se considera como normal (por exemplo 20°, como obtido experimentalmente). A terceira simplificação assume corpos perfeitamente reflectores e de textura comum e propícia à reflexão — a maioria dos casos práticos verifica estas condições.

Outra simplificação foi que a precisão da medida é indiferente da distância: esta condição é geralmente verificada nos ambientes normais de aplicação dos sensores (dimensões reduzidas), e tendo em conta as especificações do fabricante (1% de erro para toda a gama de medição). Considera-se ainda que o espaço de trabalho é bidimensional: a altura (cota) dos obstáculos é ignorada, porque simplesmente se assumem as especificações do problema (obstáculos com mais de 25 cm de altura por exemplo, são detectados a qualquer distância, mas obstáculos com 3 cm de altura não são detectados quando a menos de 2.5 metros do sensor na configuração instalada no robot).

Estas simplificações não afectam a validade da simulação dado que as situações reais estão mais ou menos confinadas a estas restrições. O mesmo não se poderá dizer dos efeitos de interferência (que todavia podem ser evitados na prática), e das reflexões especulares, que são impossíveis de evitar completamente em ambientes com paredes, armários e outras superfícies planas de extensão superior ao comprimento de onda do feixe de ultra-som (isto é, quase tudo!).

Para simplificar o problema de simulação, resolveu ignorar-se a existência de qualquer reflexão especular (que é o desejado, afinal). É ainda preciso ter em conta que o feixe de ultra-som é cónico (consequentemente em forma de sector circular num corte pelo plano paralelo ao chão, e colocado à cota dos sensores) donde é necessário, em teoria, testar toda uma frente de onda e não apenas um feixe perfeitamente linear que segue o eixo longitudinal da direcção de emissão do sensor.

Deste modo, o algoritmo para a simulação das medidas dos sensores de ultra-sons dentro de um dado ambiente foi implementado como descrito abaixo.

Definições preliminares:

$P(x,y,\theta)$ = posição do robot no referencial do ambiente,

$S_i(x_i,y_i,\theta_i)$ = disposição geométrica do sensor i relativamente ao robot (θ_i é a orientação do feixe),

$\Re$ = descrição do ambiente (conjunto de N segmentos de recta $\Re_i$),

α = abertura do feixe de ultra-sons e

β = resolução angular usada para o cálculo numérico.

D.1. Método dos raios-vector

A determinação da distância do sensor j ao ambiente será a menor das distâncias a todos os segmentos de recta do ambiente $\Re$ que se intersectam com um dos raios-vector do feixe de emissão.

O teste de intersecção com todos os segmentos do ambiente é reduzido pela distribuição desses mesmos segmentos em quadrantes (Q_1, Q_2, Q_3, Q_4) do ponto de vista do sensor e da orientação do seu eixo.

Eis o esquema base do algoritmo:

for each sensor S_j
do
  sensor_distance = **MAX_DISTANCE**
  for each direction d_k **in list** ($-\alpha/2 + k*\beta$)
  do
    temp_distance = **MAX_DISTANCE**
    for all N segments $\Re_i$
    do
      if *quadrant*(segment $\Re_i$) **not same as** *quadrant*(direction d_k) **then continue**
      else temp_distance = ***min***(temp_distance, distance to segment $\Re_i$)
    done
    sensor_distance = ***min***(temp_distance, sensor_distance)
  done
done

Este algoritmo tem custos de computação inversamente proporcionais ao valor do ângulo β. Na implementação numérica, valores muito pequenos para β (~0.5°) resultam num grande número de iterações para varrer por completo o ângulo α (~20°), e valores muito elevados (por exemplo 5°) fazem correr o risco de perder intersecções com obstáculos de dimensões menores. Tendo em conta a capacidade de cálculo disponível para a simulação (Cf. Apêndice sobre **Descrição das infra-estruturas e ambiente de trabalho**) resolveu-se usar um valor de 2° para β, que corresponde já a uma resolução angular suficiente para em simulação detectar um obstáculo de 10 cm ou mais quando colocado a 3 metros ou menos, situação perfeitamente satisfatória nos ambientes em causa.

O estratagema de avaliar o quadrante de um segmento antes de testar a intersecção com cada raio vector partindo do sensor em direcção ao infinito evita testes desnecessários acelerando desta forma o cálculo.

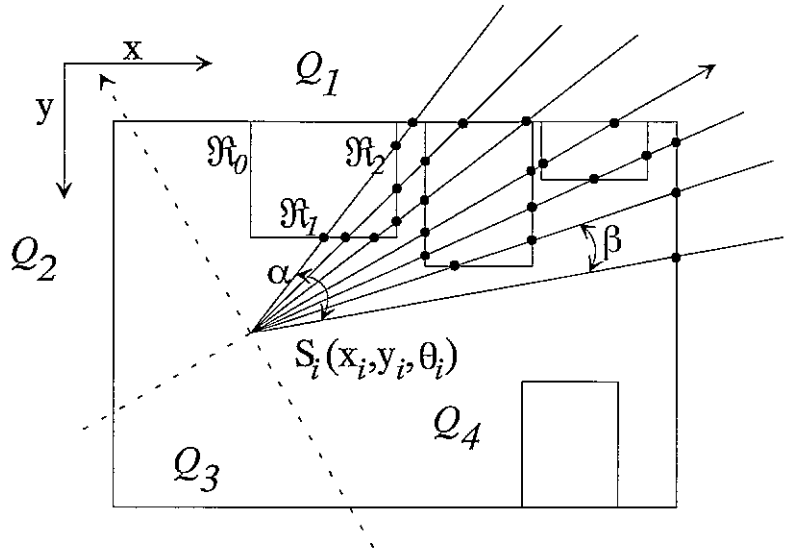


Fig. D.1 - Determinação da intersecção de um feixe de ultra-som com o ambiente.

D.2. Método dos arcos de circunferência

Uma alternativa que colmataria o problema da limitada resolução angular seria determinar a intersecção da frente de onda (arco de circunferência) com o ambiente, e dos pontos de intersecção (de 0 a 2 por cada segmento do ambiente) escolher o mais próximo do sensor. Este método seria contudo computacionalmente mais intenso, mesmo que se aproximasse o arco de circunferência por um certo número de cordas. Aqui, a grandeza contínua a discretizar para o cálculo numérico seria a distância a medir pelo sensor, e seria este a trazer a maior contribuição para o aumento das operações a efectuar. Dessa forma definir-se-iam os limites inferior e superior da distância a testar bem como um intervalo δ que seria um indicador do erro máximo para essa mesma distância. Para grandes precisões (δ muito pequenos) o número de operações crescerá proporcionalmente.

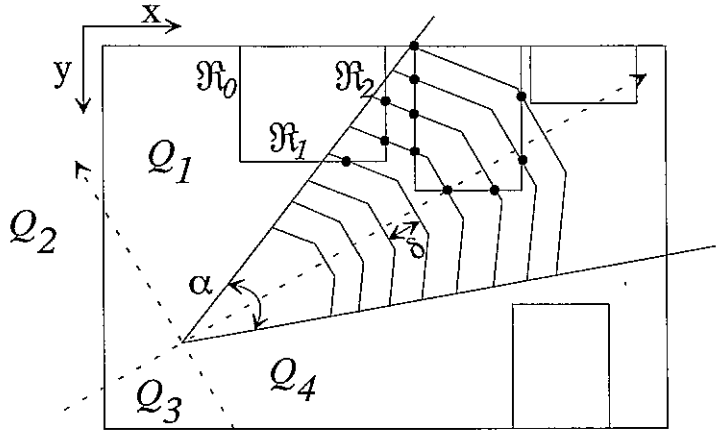


Fig. D.2 - Determinação da intersecção de um feixe de ultra-som com o ambiente usando o método dos arcos de circunferência (aqui aproximados por linhas poligonais - cordas).

Como foi dito, o método dos arcos de circunferência (ou a aproximação com conjuntos de cordas de circunferência) teria a vantagem de corresponder a uma resolução angular infinita, mas apresentaria o problema da discretização das distâncias, e o consequente aumento do número de operações como se verá de seguida.

D.3. Comparação dos métodos dos raios-vector com o dos arcos de circunferência

Estes dois métodos apresentam um diferença fundamental que corresponde a precisões (consequência da estipulação de uma dada resolução) de grandezas diferentes. O método dos raios-vector tem resolução limitada na localização angular, e o método dos arcos de circunferência tem resolução limitada na localização linear.

As exigências de cálculo podem-se avaliar pelo número de intersecções geométricas a testar. O caso dos arcos de circunferência recorre a intersecções dum segmentos de recta (o ambiente) com uma curva de 2ª ordem (o arco), mas que poderia ser aproximado por um determinado número de segmentos (cordas) caso a intersecção de 2ª ordem se tornasse demasiado exigente em termos de cálculo. O número de intersecções a testar é dado pela seguinte expressão:

$$\text{Número de intersecções} = \text{Número de segmentos do ambiente} \times \frac{\text{Gama completa}}{\text{Resolução}}$$

Como se pode ver pela tabela abaixo, o método dos raios-vector exige muito menos operações para circunstâncias consideradas normais. Note-se contudo que estes valores são algo ilusórios dado que o algoritmo dos arcos de circunferência tem prestações que dependem da distância dos obstáculos: se o algoritmo seguir uma ordem crescente para as distâncias, as intersecções com os obstáculos a distâncias curtas levarão menos operações para serem determinadas. Seguir uma ordem decrescente, começando portanto pelas longas distâncias, não apresentaria vantagens em termos de cálculo uma vez que a distância de um sensor ao ambiente é a menor das distâncias de intersecção, forçando sempre o teste das distâncias menores. Desta forma, e para o caso geral, este algoritmo não apresenta vantagens em termos de número de operações quando comparado com o método dos raios-vector, cuja eficiência é constante e independente da distância do sensor ao ambiente.

Método	Grandeza a discretizar para o cálculo (Cf. texto)	Resolução típica de interesse	Gama completa	N° máximo de intersecções a testar para um ambiente com 5 quadriláteros
Raios-vector	Ângulos do feixe	2°	20°	200
Arcos de circunferência	Distâncias a medir	5 cm	15 a 400 cm	1540 (arcos) 4620 (aprox. a 3 segm.)

Tabela D.1- Comparação dos métodos para a determinação das intersecções no simulador de ultra-sons.

Apêndice E - Um sistema de emergência usando ultra-sons

Um sistema de navegação com uma componente de autonomia, ou apenas com funções de assistência a teleoperação humana carece necessariamente de meios próprios que evitem ou minimizem o risco de colisão. Neste trabalho, e possivelmente em muitos outros de índole similar, os sensores de ultra-som propiciam-se também, apesar das suas limitações (velocidade da medição, fraca resolução angular), a uma função de detecção de emergências, isto é, de situações de colisão iminente.

O princípio base adoptado é bastante simples: existe risco de colisão quando um determinado número de sensores contíguos geometricamente acusam uma medida de distância inferior a um valor pré-estabelecido.

Este princípio foi depois estendido de forma a minimizar as falsas detecções de emergências.

O algoritmo de detecção tem um parâmetro que é o número mínimo de sensores a considerar para que haja uma situação de emergência. Qualquer valor de 1 ao número total de sensores é válida, mas os valores mais interessantes na prática são: 2 sensores consecutivos ou simplesmente apenas 1 sensor. O caso de um único sensor oferece a maior segurança possível mas não é imune a eventuais falhas de medição (ajuste do sensor ou esporádico efeito de interferência de outros sensores) detectando deste modo falsas situações de emergência.

Outros elementos do algoritmo incluem o facto da distância crítica ser dependente da velocidade instantânea do robot: quando maior a velocidade, maior deverá ser a distância crítica. A distância crítica é também, necessariamente, função da taxa sensorial: para uma taxa que varia entre duas a três amostras por segundo, deve-se tomar em consideração o caso mais desfavorável. Deve-se ainda acrescentar que o tempo de paragem (mesmo brusca) varia com a velocidade e até com o tipo de piso, donde seja muito difícil calcular teoricamente os valores óptimos para as distâncias críticas. Os valores usados, que foram corrigidos (baixados) pela experiência, seguem listados na tabela abaixo e foram repartidos em escalões por questões práticas:

Velocidade (cm/s)	*Distância (cm)*
10-20	15
20-40	25
40-60	40
60-80	50

Tabela E.1- Velocidades e distâncias de emergência (a resolução em velocidade é de 10 cm/s)

Os sensores de ultra-som encontram-se distribuídos de forma regular em torno do veiculo formando um anel, de tal forma que ao último sensor (nº 24) segue o sensor número 1. A distribuição é tal que as direcções normais do movimento (avançar/recuar) contam com maior densidade de sensores, ficando os lados do robot contemplados com um número mais

reduzido de sensores, situação justificada em parte pelas questões **holonómicas** inerentes a esta geometria de veículo: não há movimentos laterais.

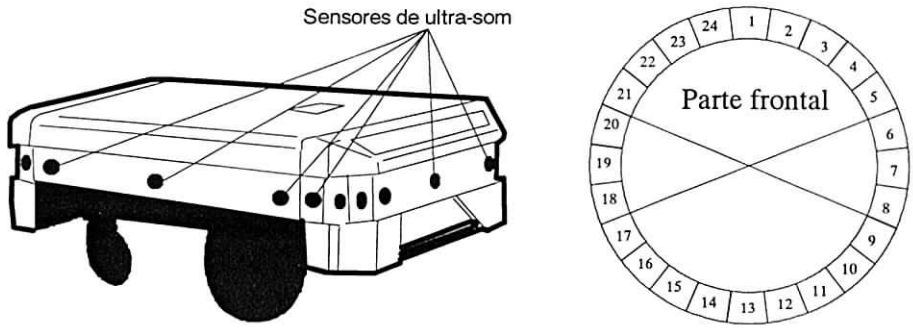


Fig. E.1 - Distribuição dos sensores de ultra-som em torno do robot. À direita uma representação esquemática da mesma distribuição e com a divisão de sensores por direcções vistas do veiculo.

O algoritmo de detecção de emergência entra igualmente em consideração com o movimento instantâneo do robot; se por exemplo o movimento em execução for para a frente, situações de curtas medições (potenciais emergências) em sensores da retaguarda não devem ser considerados emergências e levar consequentemente o robot à paragem. Esta questão aparentemente simples de desprezar medidas de sensores para forçar a paragem do robot pode-se tornar mais complexa. Um exemplo de ocorrência comum é o caso de o robot avançar tanto que se aproxima além dos limites de medição mínima ao ambiente: aí só restará a possibilidade de rotação (velocidade das rodas motrizes simétrica). Neste caso, e para que a rotação não seja erradamente interrompida ou impedida, os sensores frontais não podem ser considerados para a detecção de emergência enquanto a manobra não terminar.

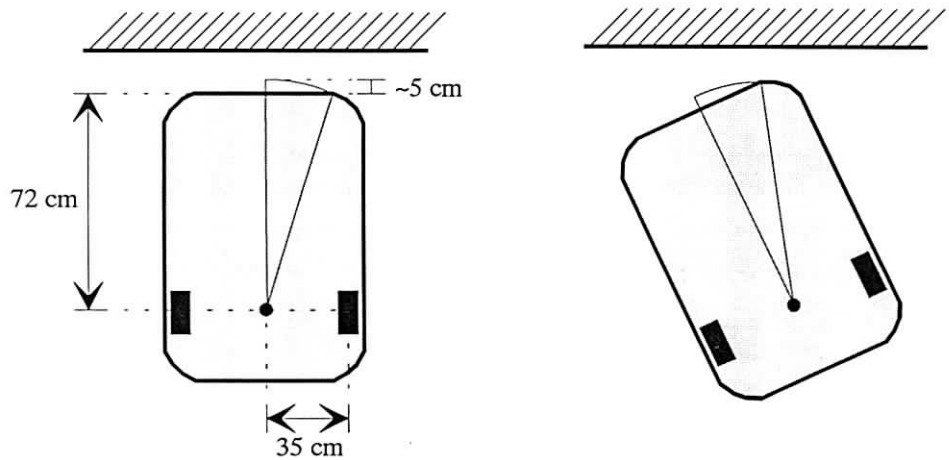


Fig. E.2 - Rotação com frente obstruída. A desinibição dos sensores frontais é necessária e não envolve risco dada a existência de uma distância de paragem de emergência suficientemente grande.

Nesta situação, o risco de contacto com o ambiente enquanto se roda é mínimo porque, aquando da aproximação, a paragem ocorreu a uma distância mínima de 15 cm (Cf. Tabela anterior). Como se pode ver pela figura abaixo, a rotação em torno do eixo normal ao eixo das rodas pode ser efectuada sem monitorização das distâncias frontais.

Os passos principais do algoritmo de detecção de emergência com base nos sensores de ultra-som são:

- Análise em anel das 24 medidas de ultra-som.
- Detecção de todas as possíveis sequências de sensores com valores inferiores ao valor crítico.
- Rejeição ou aceitação das sequências detectadas em função da direcção de movimento.

Os parâmetros do algoritmo são a **distância crítica de paragem** (função da velocidade do robot) e o **número mínimo de sensores** que definem uma sequência de emergência.

```
for all sets within the 24 sensors
do
   test set_of_consecutive_sensors
   if (emergency sequence found) then
      if ( (sensors_of_sequence on motion_direction) or ignore_motion_direction ) then
         emergency = yes
         stop robot
         set stop_cause
         return
      else continue
      endif
   else continue
   endif
done
```

Referências

[Almeida87] L. B. Almeida - Backpropagation in Perceptrons with Feedback, *Neural Computers*, 1987.

[Almeida87b] L. B. Almeida - A learning rule for assynchronous perceptrons with feedback in a combinatorial environement, *ICNN-87*, S. Diego, June 1987.

[Anderson88] J. A. Anderson, Rosenfeld - Neurocomputing, *MIT Press*, 1988.

[ANNIE91] ANNIE - Applications of Neural Networks for Industry in Europe - ANNIE, *Harwell Laborotory*, ANNIE Consortium, November 1991.

[Arkin94] R. C. Arkin, D. MacKenzie - Temporal Coordination of Percetual Algorithms for Mobile Robot Navigation, *IEEE Trans. on Robotics and Automation*, vol. 10, n. 3, pp. 276-286, June 1994.

[Ayache89] N. Ayache, O. D. Faugeras - Maintaining Representations of the Environment of a Mobile Robot, *IEEE Trans. on Robotics and Automation*, pp. 804-819, 1989.

[Barraquand91] J. Barraquand, J. C. Latombe - Robot Motion Planning: A Distributed Representation Approach, *International Journal of Robotics Research*, vol. 10, n. 6, pp. 628-649, December 1991

[Baum88] Eric B. Baum , David Haussler - What Size Net Gives Valid Generalization?, *NIPS-88*, 1988.

[Bekey93] George A. Bekey, Kenneth Y. Goldberg - Neural Networks in Robotics, *Kluwer Academic Publishers*, 1993.

[Borenstein88] J. Borenstein, Yoram Koren - Obstacle Avoidance with Ultrasonic Sensors, *IEEE Journal of Robotics and Automation*, vol. RA-4, n. 2, April 1988.

[Borenstein89] J. Borenstein , Yorem Koren - Real-Time Obstacle Avoidance for Fast Mobile Robots, *IEEE trans. on Systems*, Man and Cybernetics, vol. 19, n. 5, September/October 1989.

[Borenstein91] J. Borenstein, Yoram Koren - The Vector Field Histogram-Fast Obstacle Avoidance for Mobile Robots, *IEEE trans. on Robotics and Automation*, vol. 7, n. 3, June 1991.

[Borenstein92] J. Borenstein, Y. Koren - Noise Rejection for Ultrasonic Sensors in Mobile Robot Applications, *Proc. of the IEEE Int. Conf. on Robotics and Automation - Nice*, France, May 1992.

[Borenstein95] J. Borenstein, Y. Koren - Error Eliminating Rapid Ultrasonic Firing for Mobile Robot Obstacle Avoidance, *IEEE Trans. on Robotics and Automation*, Vol. 11, n. 1, pp.132-138, Feb. 1995.

[Brooks83] R. A. Brooks, T. Lozano-Pérez - A Subdivision Algorithm in Configuration Space for Findpath with Rotation, *Proc. of 8th Conf. on Artificial Intelligence*, Karlsruhe, Germany, pp. 799-806, 1983

[Brooks86] R. A. Brooks - A Robust Layered Control System for a Mobile Robot, *IEEE Journal of Robotics and Automation*, vol. 2, n. 1, Mar 1986, pp.14-23.

[Canny88] J. F. Canny - The complexity of Robot Motion Planning, *MIT Press*, Cambridge, MA, 1988.

[Clarke93] Roger Clarke - Asimov's Laws of Robotics: Implications for Information and Technology (Part 1), *IEEE Computer*, pp.53-61, December 1993.

[Clarke94] Roger Clarke - Asimov's Laws of Robotics: Implications for Information and Technology (Part 2), *IEEE Computer*, pp.57-66, January 1994.

[Connell93] J. H. Connell, S. Mahadevan - Robot Learning, *Kluwer Academic Publishers*, 1993.

[Cox91] I. J. Cox - Blanche-An Experiment in Guidance and Navigation of an Autonomous Robot Vehicle, *IEEE Trans. on Robotics and Automation*, vol. 7, n. 2, April 1991.

[Crowley85] J. L. Crowley - Navigation for an Intelligent Mobile Robot, *IEEE Journal of Robotics and Automation*, vol. RA-1(1), pp. 31-41, March 1985.

[Crowley89] James L. Crowley - World Modeling and Position Estimation for a Mobile Robot Using Ultrasonic Ranging, *IEEE Int. Conf. on Robotics and Automation*, Scottsdale, May 1989.

[Dayhoff90] Judith Dayhoff - Neural Networks Architectures - An Introduction, *Van Nostrand Reinhold*, 1990.

[Donald87] B. R. Donald - A Search Algorithm for Motion Planning with Six Degrees of Freedom, *Artificial Intelligence*, vol. 31 (3), pp. 295-353, 1987.

[Drumheller87] Michael Drumheller - Mobile Robot Localization Using Sonar, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 9, n. 2, March 1987.

[Elfes89] Alberto Elfes - Using Occupancy Grids For Mobile Robot Perception and Navigation, *IEEE Computer*, pp. 46-57, June 1989.

[Feng90] Dai Feng, B. H. Krogh - Satisficing Feedback Strategies for Local Navigation of Autonomous Robots, *Autonomous Mobile Robots (Volume 1)*, IEEE Press, pp. 476-488, 1990.

[Feng92] Liqiang Feng, Yeshaiahu Fainman, Yoram Koren - Estimation of the Absolute Position of Mobile Systems by an Optoelectronics Processor, *IEEE Trans. on Systems*, Man, and Cybernetics, vol. 22, n. 5, pp. 953-963, Sep/Oct 1992.

[Feng94] L. Feng, J. Borenstein, H. Everett - Sensors and Methods for Autonomous Mobile Robot Positioning, *University of Michigan Technical Report UM-MEAM-94-21*, December 1994.

[Gonçalves91] J. Gonçalves, P. Veiga, G. Campos, J. Perdigão, F. Sorel - Navigation of a Tele-operated Vehicle in a Safeguards Environment, *Proc. of 13th ESARDA Symposium on Safeguards and Nuclear Material Management*, Avignon, France, May 1991.

[Gonçalves93] J. Gonçalves, G. Campos, V. Santos, V. Sequeira, F. Silva - Mobile Robotics Applied to the Remote Verification of Storage Areas, *Proc. of 15th ESARDA Symposium on Safeguards and Nuclear Material Management*, Rome, May 1993.

[Gonçalves93b] J. Gonçalves, G. Campos, V. Santos, V. Sequeira, F. Silva - Mobile Robotics for the Surveillance of Fissile Materials Storage Areas: Sensors and Data Fusion, *in Data Fusion Applications*, (Ed. by S. Pfleger, J. Gonçalves, D. Vernon), Springer-Verlag, pp. 214-245, 1993.

[Gorman88] R. P. Gorman , T. J. Sejnowski - Analysis of Hidden Units in a Layered Network Trained to Classify Sonar Targets, *Neural Networks*, vol. 1, 1988.

[HechtNielsen90] Robert Hecht-Nielsen - Neurocomputing, *Addison-Wesley*, 1990.

[Holenstein92] A. A. Holenstein, M. A. M"uller, E. Badreddin - Mobile Robot Localization in a Structured Environement Cluttered with Obstacles, *Proc. of the IEEE Int. Conf. on Robotics and Automation - Nice*, France, May 1992.

[Huang91] Shih-Chi Huang , Yih-Fang Huang - Bounds on the Number of Hidden Neurons in Multilayer Perceptrons, *IEEE trans. on Neural Networks*, vol. 2, n. 1, January 1991.

[Hush93] Don R. Hush, Bill G. Horne - Progress in Supervised Neural Networks, *IEEE Signal Processing Magazine*, pp. 8-39, January 1993.

[Jackel88] L. D. Jackel, H. P. Graf, W. Hubbard, J. S. Denker, D. Henderson, Isabelle Guyon - An application of neural net chips: handwritten digit recognition, *Neurocomputing 2*, MIT Press, 1988.

[Kathib85] O. Kathib - Real-Time Obstacle Avoidance for Manipulators and Mobile Robots, *Proc. of the IEEE Int. Conf. on Robotics and Automation*, St. Louis, May 1985, pp. 500-505.

[Kinsler62] Lawrence E. Kinsler, Austin R. Fey - Fundamentals of Acoustics, *John Wiley & Sons*, Inc., 1962.

[Kleeman92] Lindsay Kleeman - Optimal Estimation of Position and Heading for Mobile Robots Using Ultrasonic Beacons and Dead-reckoning, *Proc. of the IEEE Int. Conf. on Robotics and Automation - Nice*, France, May 1992.

[Kohonen89] T. Kohonen - Self-Organization and Associative Memory, *Springer-Verlag*, May 1989.

[Kriegman89] D. J. Kriegman, E. Triendl, T. O. Binford - Stereo Vision and Navigation in Building Mobile Robotics, *IEEE Trans. on Robotics and Automation*, vol. 5 (6), pp. 792-803, 1989

[Krogh84] B. H. Krogh - A Generalized Potential Field Approach to Obstacle Avoidance Control, *Robotics International Robotics Research Conference*, Bethlehem,PA, August 1984.

[Krogh86] Bruce H. Krogh , Charles E. Thorpe - Integrated Path Planning and Dynamic Steering Control for Autonomous Vehicles, *Proc. IEEE Int. Conf. on robotics and Automation*, pp. 1664-1669, 1986.

[Krogh89] B. H. Krogh, Dai Feng - Dynamic Generation of Subgoals for Autonomous Mobile robots Using Local Feedback Information, *Autonomous Mobile Robots (Volume 1)*, IEEE Press, pp. 465-475, 1989.

[Latombe91] Jean-Claude Latombe - Robot Motion Planning, *Kluwer Academic Publishers*, 1991.

[LeCun90] Y. LeCun, B. Boser, D. Henderson, J. S. Denker, R. E. Howard, W. Hubbard, L. D. Jackel - Backpropagation applied to handwritten zip code recognition, *Neurocomputing 2*, MIT Press, 1990.

[Leonard90] J. Leonard, H. Durrant-Whyte, I. J. Cox - Dynamic Map Building for an Autonomous Robot, *Autonomous Mobile Robots (Volume 1)*, IEEE Press, pp. 331-338, 1990.

[Leonard92] John J. Leonard , Hugh F. Durant-Whyte - Directed Sonar Sensing for Mobile Robot Navigation, *Kluwer Academic Publishers*, 1992.

[Lippmann87] R. P. Lippmann - An Introduction to Computing With Neural Nets, *IEEE ASSP Magazine*, April 1987.

[LozanoPerez81] T. Lozano-Pérez - Automatic Planning for Manipulator Transfer Movements, *IEEE Trans. on Systems*, Man and Cybernetics, SMC-11 (10), pp. 681-698, 1981.

[LozanoPerez83] T. Lozano-Perez - Spatial Planning: A Configuration Space Approach, *IEEE Transactions on Computers*, vol C-32, n. 2, pp.108-120, 1983.

[McKerrow93] P. H. McKerrow - Introduction to Robotics, *Addison-Wesley*, 1993.

[McKerrow93b] P. J. McKerrow - Echolocation: From Range to Outline Segments, *Proc. Int. Conf. on Intelligent Autonomous Systems*, Pittsburg, PA, pp.239-247, 1993.

[Millan92] J. R. Millan - Building Reactive Path-Finders through Reinforcement Connectionist Learning: Three Issues and An Architecture, *Proc. 10th European Conf. on Artificial Intelligence*, Vienna, Austria, 1992.

[Mirchandani89] Gagan Mirchandani , Wei Cao - On Hidden Nodes for Neural Nets, *IEEE trans. on Circuits and Systems*, vol. 36, n. 5, May 1989.

[Moita94] F. Moita, A. Feijão, U. Nunes, A. T. Almeida - Modelling and Calibration of an Ultrasonic Ranging System of a Mobile Robot, *1st Portuguese Meeting on Automatic Control*, CONTROLO94, Lisbon, September 1994.

[Moore92] Kevin L. Moore - Artificial Neural Networks, *IEEE Potentials*, Feb 1992.

[Moravec85] Hans P. Moravec, Alberto Elfes - High Resolution Maps from Wide Angle Sonar, *Proc. IEEE Int. Conference on Robotics and Automation*, Washington D.C., pp. 116-121, March 1985.

[NationalInst92] National Instruments, Product Catalog, 1992.

[Nilsson69] N. J. Nilsson - A Mobile Automaton: An Application of Artificial Intelligence Techniques, *Proc. of the 1st Int. Joint Conference on Artificial Intelligence*, pp. 509-520,Washington D.C., 1969.

[Noborio90] H. Noborio , K. Kondo , A. Noda - A Good extension Method of Free Space in an Uncertain 2D Worksapce by using an Ultrasonic Sensor, *IEEE Proceedings IROS'90*, pp. 665-671, 1990.

[ODunlaing85] C. O'Dunlaing, C. K. Yap - A Retraction Method for Planning the Motion of a Disc, *Journal of Algorithms*, n. 6, pp. 104-111, 1982.

[Pedrosa94b] F. Pedrosa, M. Isabel Ribeiro - Sonar Data Processing for the Navigation of a Mobile Robot, *5th Int. Conf. on Signal Processing Applications & Technology*, ICSPAT94, Dallas,USA, October 1994.

[Pomerleau93] Dean A. Pomerleau - Neural Network Perception for Mobile Robot Guidance, *Kluwer Academic Publishers*, 1993.

[Preparata85] F. P. Preparata, M. I. Shamos - Computational Geometry, *Springer-Verlag*, 1985.

[Robosoft91b] ROBOSOFT - Robuter Users's Manual, *V3.1*, August 1991.

[Rumelhart86] D. E. Rumelhart , G. E. Hinton, R. J. Williams - Learning internal representations by error propagation, *Neurocomputing*, MIT Press, 1986.

[Santamaria94] Juan C. Santamaria , R. C. Arkin - Model-based Echolocation of Environmental Objects, *IROS 94*, Munich, Germany, September 1994.

[Santos93] V. Santos, J. Gonçalves, F. Vaz - Perception Maps for the Local Navigation of a Mobile Robot using Ultrasound Data Fusion, *Esprit Workshop on Multisensor Applications*, November 1993.

[Santos93b] V. Santos, J. Gonçalves, F. Vaz - Ultrasound Sensors for Environment Perception, *JRC Technical Note No. I.93.128*, October 1993.

[Santos94] V. Santos, J. Gonçalves, F. Vaz - Perception Maps for the Local Navigation of a Mobile Robot: a Neural Network Approach, *IEEE Int. Conf. on Robotics and Automation*, San Diego, USA, pp. 2193-2198, May 1994.

[Schwartz83] J. T. Schwartz, M. Sharir - On the Piano Mover's Problem: I. The Case if a Two-Dimensional Rigid Polygonal Body Moving Amidst Polygonal Barriers, *Communications on Pure and Applied Mathematics*, vol. 36, pp345-398, 1983.•

[Sedgewick84] Robert Sedgewick - Algorithms, *Addison-Wesley*, 1984.

[Sejnowski86] T. J. Sejnowski, C. R. Rosenberg - NETtalk: a parallel network that learns to read aloud, *Neurocomputing*, MIT Press, 1986.

[Sequeira92] Vitor Sequeira, João Gonçalves - A Graphical Environment for the Navigation of a Mobile Robot, *Proc. of Compugraphics 92*, Lisbon, pp. 336-354, December 1992.

[Sequeira93] Vitor Sequeira, João Gonçalves - Control and Navigation of a Mobile Robot, *JRC Technical Note No. I.93.137*, October 1993.

[Sequeira93b] Vitor Sequeira, João Gonçalves - Position Estimation of a Mobile Robot using Laser Range Data, *JRC Technical Note No. I.93.132*, October 1993.

[Sharir89] Micha Sharir - Algorithmic Motion Planning in Robotics, *IEEE Computer*, pp. 9-19, March 1989.

[Silva92] Filipe Silva, João Gonçalves - A Graphical Tool for the Tele-operation of a Manipulator Arm, *Proc. of Compugraphics 92*, Lisbon, pp. 346-354, December 1992.

[Takahashi89] Osamu Takahashi , R.J. Schilling - Motion Planning in a plane Using Generalized Voronoi Diagrams, *IEEE trans. on Robotics and Automation*, vol. 5, n. 2, April 1989.

[Tilove90] Robert B. Tilove - Local Obstacle Avoidance for Mobile Robots Based on the Method of Artificial Potentials, *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 566-571, 1990.

[Vogl88] T. P. Vogl, J. K. Mangis, A. K. Rigler, W. T. Zink, D. L. Alkon - Accelerating the Convergence of the Back-Propagation Method, *Biological Cybernetics*, vol. 59, pp. 257-263, 1988.

[Widrow88] Bernard Widrow , R. G. Winter , R. A. Baxter - Layered Neural Nets for Pattern Recognition, *IEEE trans. on Acoustics Speech ans Signal Processing*, vol. 36, n. 7, July 1988.

[Wylie82] C. Wylie, L. Barrett - Advanced Engineering Mathematics, *McGraw-Hill*, 1982.

[Zelinsky88] A. Zelinsky - Environment Mapping with a Mobile Robot Using Sonar, *Proc. of the Australian Joint Artificial Intelligence Conference - AI 88*, pp. 373-388, November 1988.

[Zurada92] Jacek M. Zurada - Introduction to Artificial Neural Systems, *West Publishing Company*, 1992.